Lost in SPACES: Exploring the Benefits and Shortcomings of Spatial Presence and Awareness as a Mechanism for Context Reasoning

Sara Coverdale

Abstract

Context-Aware applications make use of sensed and gathered information about a user's state to better tailor their behaviour to the user's needs. There are many streams of information that can be employed as context; these elements have a variety of structures and not all of them are static or fully known to developers at runtime, this can make it challenging to add new streams of context to an application and keep those streams whose structure frequently changes updated. Heterogeneity of sensor technology and information sources means two users may generate information about the same aspect of their state, their location for example, in two different formats. Although there are examples of services that can relate a value from one location sensing technology to a value from another, we lack a general service for building and reasoning about these relationships between any and all representations of context. Moreover, due to their frequent use of sensed physical information, context-aware applications also generate seams, which may cause uncertainty, error and unexpected behaviours. Developers need a simple way to think about context; one which supports the many heterogeneous types of information it comprises, allows them to update the structures of those types as they change or are discovered, gives them the means to reason about relationships between users whose devices sense the same sort of context using different technologies, and which provides them with both opportunity and means to address and respond to some of the seams arising in their applications.

This work presents a spatial model for reasoning about context using presence and awareness that attempts to address the requirements above. The model is realised in a platform comprised of an API and a Context-Broker, which provides context-reasoning and relation services for applications that communicate with it via a dataspace. Using the platform to implement a location-aware game raises several issues concerning the usability and presentation of its paradigms and a seam encountered by a user playing the game. We use this as an opportunity to explore methods for analysing how underlying models, platforms and applications create meaning in pervasive and Mixed-Reality context-aware applications, and how seams operate within

them. We present changes to our model and platform based on the conclusions drawn from this analysis. Our analysis of the user's experience leads us to conclude that model paradigms must be carefully structured to support the multiple ways in which context is used, that both spatial and temporal discontinuity can create user confusion and that when mixing virtual and physical spaces it is important to communicate fully all the structures imposed upon them. Our analysis of seams leads us to the conclusion that responses to them can be usefully positioned as simple processes of disparity resolution involving the restructuring of one or more spaces or milieu involved in the application.

Table of Contents

Table of Figures

Chapter 1

Introduction

Background

Context-Aware applications leverage information about a user's state to tailor their behaviour to better serve the user's needs. Context can encompass myriad properties of a user's state, both sensed and constructed, and many of these properties can be gathered in multiple ways. A user's location for example can be sensed via GPS or GSM. A user's social connections too, may vary in content and structure dependent on which social network the data is taken from. Whilst Facebook's [FACEBOOK, '16] friendship connections are mutual, for example, those of Twitter [TWITTER, '16] are asymmetric—a user can follow people who do not follow her and vice versa. In addition to the possible types and formats of context a developer may choose to work with, constraints arising from the technologies available to a user must also be considered. Many laptops, for example, offer no GPS or GSM support, and devices may be affected by different communications, processing and power limitations, which developers must take into account. Pervasive and Mobile application developers are often faced with the question of how to support and reconcile the capabilities of such heterogeneous technologies.

Whilst some forms of context like GPS are well defined and understood, others, especially those that leverage infrastructures not under the developer's control, GSM for example, must be explored and mapped before they can be used. Still others are partly or entirely dynamic, presenting new structures and sets of possible user states over time. Social networks change as new people join or leave them, just as a Wi-Fi positioning space can change as new SSID's are set-up and retired in the area it covers. If a developer wishes to make use of two more forms of a Context property, to allow users of one to interact with those of another under their shared context, she must understand how a user's

state in one form relates to a user's state in the other. This understanding of the dynamic forms of context and their relationship to each other can be difficult for individual developers to achieve alone. Projects like Placelab allow developers to collect war-driven data[1] on location infrastructures like GPS, GSM and Wi-Fi, but even these rely both on a significant investment of time and resources on the behalf of developers before they can include a new technology in their application and significant ongoing effort to continually detect and integrate updates from changing context structures. Furthermore, these projects require that every application explicitly understand the structure of every type of context it uses. When a new type of context is introduced, a new way to sense location for example, developers wishing to use it must rewrite their applications to take account of it. Simply put, how a user's context can be used is dependent on how it is gathered to an extent which places undue burden on the developers of context-aware applications, one of this work's aims is to enable the two processes' separation.

Complexity in Mixed-Reality Pervasive Applications

Context-Aware applications are often also Pervasive, Mixed-Reality applications. Applications of this type combine information drawn from the sensed physical environment, constructed virtual worlds, and users' personal and social histories of the spaces they engage with, operating for their users as a complex interweaving [CHALMERS and GALANI'04] of many different milieus: physical, technological, virtual, social, personal, and historic amongst them. This makes them difficult to analyse according to traditional practices of application development. Questions such as was this behaviour correct? Why not? What caused its outcome? Quickly become complicated with multiple intersectional factors; the effect of tall buildings on the accuracy of a GPS signal and the user's ability to understand and respond to that effect is one of the less complex examples of this. Chalmers and MacColl name the often

[1] This is data collected through a brute-force process of visiting every physical location and recording the GPS coordinate and/or WiFi, GSM or other signals sensed at that location. The sensed data can then be uploaded to a central server for use by developers in their location-aware apps. The name comes from a combination of War dialling, the technique of using a modem to scan a list of telephone numbers for accessible computers, bulletin boards or fax machines, and the fact that data is often collected by driving around the chosen area.

problematic result of these intersections “Seams”[CHALMERS and GALANI'04] . They argue that where developers traditionally ignore or hide the problems arising at such sites of interaction, or interface between different milieu, Pervasive Mixed-Reality with its complex and often unexpected processes of intersection at those sites, invites a different approach: to allow users and applications to appropriate such Seams, turning them into tools or aspects of gameplay. An additional aim of our work is to provide Context-Aware application developers with ways in which to think more easily about these complex Mixed-Reality spaces and their Seams.

Research Focus

Our research concerns itself with the construction of a model to support applications in reasoning within and across heterogeneous and dynamic domains of context where behaviour within the application can be structured as interaction between *Producers* and *Consumers* of information, services or behaviours. As we show in Chapters 3 and 4, this structure of behaviour covers a wide range of application scenarios. The model provides structures for abstracting many different types of context to simple spatial forms and offers processes for reasoning upon user states within and across these spaces. We realise the model as a platform, comprised of a Context-Broker application and API for accessing it, which we call the Spatial presence and Awareness Context Evaluation Service (SPACES).

The SPACES model and platform were used to develop and demonstrate a Mixed-Reality Context-Aware game, called Bombsquad. Here the same complex intersectional factors discussed in the previous section gave rise to some questions concerning the usability of the model. These demonstrated that, even at the theoretical and platform level, Pervasive systems can find it difficult to isolate themselves from the often seam-driven effects of interaction between different milieu that so characterises their spaces of operation. These issues directed our focus slightly away from Computer Science as we explored how relevant architectural work on the experience of Pervasive spaces and philosophical work on technology as process, can be combined and employed to help us better understand the mechanisms at work in these systems. From

this understanding we derive and present changes to improve the SPACES model and platform, and the Bombsquad application.

Research Aims

Pervasive and Mixed-Reality systems are a tangle of interactions and relationships between different domains, spaces and milieu. As a subset of these systems, Context-Aware applications also suffer from this complexity and heterogeneity, which does not only extend to how different aspects and representations of user context relate to each other, nor how entities might be thought of as interacting with and within those dynamic and often partially known spaces, but reaches beyond these into interactions of personal and social history with physical space, interactions of social processes with virtual space, and many more. This research aims to provide a Model and Platform to reason about a small technological subset of these relationships, between different representations of elements of user context – where developers may still be exploring at runtime those elements' structures and relations.

Research Questions

1. Is there a way to think and reason uniformly about context that supports the heterogeneous types and forms of context that are found in context-aware applications today and may emerge in the future?
2. How can we describe context in representations whose content and structure are still being discovered, or may be subject to change, in such a way that we retain the ability to reason about those forms of context?
3. What structures and processes in a Pervasive Mixed-Reality application drive its ability to be meaningfully understood by its users?
4. How can we think about seams in terms of these structures and processes at an abstract level?
5. How can those elements be employed by developers to create platforms and models whose structures are more easily comprehended?

6. How can those elements be used by platforms in order to provide services that allow developers to mindfully manage, present or take advantage of seams?

Thesis Synopsis

Chapter 2 explores the relevant literature, which motivates our research and supports the design choices in our model and platform. We start with an examination of what context actually is, how it acts and is acted upon in applications, and how some existing applications from the literature approach the process of reasoning about it. Since our own approach to context is spatial, the next section concerns itself with how various fields, including robotics and town planning, reason about space itself. This leads us to the presence and Awareness paradigm conceived by Benford et al [BENFORD and FAHLÉN'93] for the purpose of reasoning about user interaction in virtual reality spaces and later extended by Rodden [RODDEN'96] . This, we use as a basis for our model. The presence and Awareness paradigm is driven by the idea of areas of interest and influence intersecting or overlapping in space and we contextualise this approach against the qualitative spatial reasoning system RCC-8 which provides a general approach to reasoning about the arrangement of pairs of entities in 2-dimensional space with respect to each other. Finally, since all Pervasive Systems, including our own, are affected by seams, we provide an overview of the work to both identify and exploit seams in Pervasive systems.

Chapter 3 defines our theoretical SPACES model, which takes on both Dix's Physical World-Representation-Model architecture [DIX, FRIDAY et al.'05] and the presence and Awareness paradigm as its basis. We explain how context can be thought of in terms of proximity between two entities in a space and the ways in which presence and Awareness relate to this. We give qualitative definitions of different types of Awareness arising from the various presence configurations. Finally, we describe how our model supports reasoning about Awareness not just within individual spaces, but across pairs of spaces as well.

Having described the abstract structures of our SPACES model, Chapter 4 then proceeds with a definition of the mathematical structures and processes that

comprise the model. We define 4 different types of space which map to differently structured context representations and show how the relevant features of these representations are abstracted into the structures and properties of the spaces. We show how a user's context becomes her presence in a space and how our model supports relations between spaces such that users can gain awareness of entities with differently represented context states and therefore presence in different spaces. We also explain how a user's awareness of others is calculated both in a single space and as an aggregation of awareness across pairs of spaces. One class of spaces, which we term Asymmetric-Discrete, exhibits markedly different behaviour to the others under the calculation of certain types of Awareness and we both explore this difference and offer solutions for managing it.

We implemented our model as an API and Context-Broker system called SPACES. Chapter 5 covers the architecture and implementation details of SPACES, explaining how its programmatic structures map to the structures of the model and implement its processes. The first part of the chapter concerns itself with the structures of the API, which application developers must implement to communicate the context representations they use and their users' states within those representations to the Context-Broker. The latter half of the chapter details how the Context-Broker enacts the mathematical reasoning processes of the model, given these implementations of its structures, and the processes it employs to manage the dynamic nature of the spaces it reasons upon and their relationships to each other.

Having designed SPACES to offer a reasoning service for applications using heterogeneous and dynamic context, we then constructed an application to use such a service. Chapter 6 describes the implementation of Bombsquad, a mobile location-aware game played on the Symbian S60 platform and coordinated by a collection of supporting server applications, which make use of the SPACES platform to perform their Context reasoning. During the initial usage of Bombsquad, however, our user encountered significant issues which prevented her from understanding the Mixed-Reality space constructed by the application and model, and navigating within it. We discuss the nature of these

three issues and how they arose, including one which was generated by the presence of a seam which we had not foreseen arising when we designed either SPACES or the Bombsquad application.

As we stated earlier, we believe understanding and thinking about the structures, processes and user experience of Pervasive Mixed-Reality applications to be complicated by their intersectional nature. In order to clarify the issues driving user understanding of our model, platform and application, we employed some tools of reasoning borrowed from the philosophies of technology and architecture. These clarify some of the meta-structures and processes at work in our application and how they relate to or arise from the various milieus which find their intersection in our application trial. Chapter 7 concerns itself with presenting these concepts to the technical reader, exploring their relationships to Pervasive Computing in general and their implications for our application and SPACES model and platform. We begin with an examination of how meaning in language arises through metastability of signifier-signified pairings and show how that same metastability is enacted through technology. We identify the conceptual structures and processes that drive metastability, the ways in which SPACES and Bombsquad fail to enact them, and the modifications we could make to correct this. We examine how our model's implementation of the Awareness paradigm situates Awareness as infrastructural yet paradoxically transient, how this tension creates user confusion, and how adjustments to the model can ameliorate this. Finally, we employ Gilbert Simondon's concept of transduction [SIMONDON'05] as explored by Adrian MacKenzie [MACKENZIE'02] as a reasoning device through which to approach the subject of seams in Mixed-Reality Applications. We present a survey of the properties and processes of seams found in our work and the literature from this position, which motivates further modifications to SPACES and the Bombsquad application.

Chapter 8 concludes the work with a summary of its aims, methodologies and contributions, a discussion of the findings of Chapter 7 with respect to the wider field of Context-Aware Mixed-Reality applications, and our overall conclusions as to the implications of these findings for context-aware

application developers. We close by offering our suggestions for future avenues to explore.

Chapter 2

Literature Review

Introduction

This work is primarily motivated by issues arising from context heterogeneity and dataset-evolution in the context-aware subfield of Pervasive Computing and is primarily concerned with reasoning about context with respect to these issues. Our approach to the problem draws on the work of Benford et al. concerning presence and Awareness in the fields of Computer Supported Cooperative Work and Mixed-Reality systems. Seams arise in Pervasive Computing at the edges of technologies, the boundaries between them and those boundaries that exist where different types of space meet. Our work takes an approach to context reasoning which is mindful of seams by bringing forward the set of these boundaries between the physical space and sensing infrastructure, however issues that arose during usage of our platform demonstrate there is much more work to be done regarding seam-centric approaches to context. In this chapter we begin by exploring how context has been defined in previous work and illustrate the range of interpretations of context in the literature. We examine the impact that the increasing ubiquity of sensors and mobile devices has on context-aware applications and its implications for developers – here we highlight the issues that motivate our work. We present an overview of the existing body of work concerning reasoning about context and discuss how our approach fits into this geography. Our model and framework are based on spatial reasoning, and so we therefore give a brief treatment of spatial reasoning systems focusing particularly on the evolution of Benford et. al's presence and Awareness work in the field of Cooperative Virtual Environments. Benford's work is the basis for ours and we trace its evolution from CVE tool to the field of Mobile Mixed-Reality Applications. Finally, several researchers have previously explored the effect of Seams on Mixed-Reality and Ubiquitous Computing applications and we provide an overview of this work.

What is Context

Mark Weiser envisioned ubiquitous systems as tools which are invisible to the user, defining invisible as "[a tool which] does not intrude on your consciousness; you focus on the task, not the tool." [WEISER'94] Context-aware systems approach this goal by adapting themselves and their behaviour to better support their users' tasks. There are many definitions of context, but a few have become notable:

Context in the field of Ubiquitous Computing was first defined by Schilit and Theimer in [SCHILIT and THEIMER'94] as a user's location, identity of nearby entities, and changes to those entities. In contemporary work, Schilit, Adams and Want [SCHILIT, ADAMS et al.'94] elaborate on their view of context as a necessary combination of properties, adding that they do not view location alone as context. Prior to this work, researchers at Olivetti had created the Active-Badge system [WANT, HOPPER et al.'92] to leverage real-time user location in providing a higher level of service for a call forwarding system within an office.

Brown [BROWN'95] expands on Schilit et al.'s interpretation in his Stick-E notes framework to include properties such as adjacency of other objects, temperature, computer states, time, and most interestingly: imaginary companions known as spirits. Brown argues that tasks which are environmentally independent can be contextualised by assigning them to spirits so that when a user specifies she is with a particular imaginary companion the application can provide services contextually appropriate to the spirit's designated task.

Other approaches to defining context are more rooted in the physical: Ryan et al. [RYAN, PASCOE et al.'97] use location, environment, identity and time as their definition whilst Abowd et al. [ABOWD, ATKESON et al.'97] focus solely on location and orientation of the user to produce a location-aware tour guide. Later, Dey, in his Context-Toolkit work [DEY'01] , introduced the possibility to consider the user's emotional state --- a non-physical and less environmental

aspect of context -- to a context-aware conference assistant. Both Franklin and Flaschbart [FRANKLIN and FLASCHBART'98] and Hull et al. [HULL, NEAVES et al.'97] take a more open-ended approach to context, simply characterising it as aspects of the current situation. Similarly Ward et al. [WARD, JONES et al.'97] describe it as the state of the application's surroundings. More recently, Rodden et al's work [RODDEN, CHEVERST et al.'98] restricts context to the application's settings. Many of these early definitions of context are chosen to allow the application to identify the user's current task, however, Chen et al. [CHEN, FININ et al.'03] consider the user's current activity or task to be part of their context. Abowd et al. encapsulate all of these views of context as: "any information that can be used to characterize the situation of an entity." Where "An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves." [ABOWD, DEY et al.'99] Dourish, in contrast to these approaches, argues for a different approach to context; that it is -- rather than something an application simply senses or derives – a property that is being constantly negotiated between those users populating the application [DOURISH'04] .

These popular definitions of context, and the individual definitions used by myriad context-aware applications that exist today, provide us with a multitude of options for constructing a user's context. We can therefore think of context whether it be assigned to a user, an application, and agent, or a group, as being composed of various combinations of properties which we call context elements. A context element is thus a single aspect of context, for example: location, temperature, task or time.

Heterogeneity and Dynamism of Context

Although their developers' definitions of context could be quite broad, in practice early ubiquitous computing systems' use of context was often more limited in scope, either as a result of the sensing hardware available or as a consequence of the reasoning systems applied to it. They tended to use only a

few elements of the theoretically available set and rarely employed alternative sensing technologies to collect the same types of data. Each piece of sensed information that was employed was generally drawn from a restricted, known, and fixed dataset and there were fewer options for obtaining that data. For example Infra-Red (IR) locations [ABOWD, ATKESON et al.'97] were restricted to an office building with new cells being added manually. Also, as many context reasoning approaches relied on sets of If-Then rules, the set of possible contextual states a user could take was limited and static. No context-aware applications employed Dourish's approach to context as a negotiated property.

Nowadays there are many more sensors and sources for context and those sources have much greater potential to generate datasets whose structure is dynamic. McCullough [MCCULLOUGH'04] describes how sensors have transitioned from single-use wired devices of considerable expense to cheaper, wireless devices that can be configured or configure themselves, and each other, on the fly. This offers today's developers both considerably wider choice of inputs when designing pervasive systems and a much greater challenge in constructing them to support those inputs.

As sensing technologies have evolved to provide a greater variety of representations of location, the range of other properties that they can sense has also broadened. In particular, mobile devices have begun to include a wider range of sensors such as gyroscopes, compasses and thermometers, and smart environments are now able to gather and reason about extremely nuanced data such as in Franklin and Flaschbart [FRANKLIN and FLASCHBART'98] which uses computer imaging via a classroom camera system to determine when a teacher's context requires that the room's projector be turned on and the lights dimmed. Zheng [ZHENG, ZHENG et al.'10] uses various properties of the user's environment, such as temperature, weather, and crowdedness, in a tourist activity application to provide more user- and environment-appropriate activity suggestions. When discussing context, we refer to each of these properties as an *element of context* or *context element*.

Despite Schilit's feeling that context should be more than location, many of the earliest context aware applications used just location or co-location as user

context. Context nowadays is not only more varied in the properties it encompasses, it is also situated across a significantly broader range of infrastructure types. Location data alone can be sensed using a range of technologies such as IR beacons as in Cyberguide, GPS as in Smart Sight [YANG, YANG et al.'99] , Wi-Fi as in Treasure [BROLL, BENFORD et al.'06] , GSM as in Hitchers [DROZD, BENFORD et al.'06] , Sonar via computer mic [TARZIA'11] , Powerline positioning [PATEL, TRUONG et al.'06] , or the Xerox Active Badge system [WANT, HOPPER et al.'92] . In the case of GPS, the technology used is a globally deployed and dedicated infrastructure which is designed and managed by third parties to provide a globally objective coordinate system; many other location sensing systems. Other technologies, however, such as Tarzia's sonar, need no such infrastructure, but create coordinate reference systems local to a device owned and controlled by the user. Others still, those such as Wi-Fi and GSM, make use of existing infrastructures that have been widely deployed for non-locational purposes and whose ownership is distributed across multiple third parties and is subject to the control of neither user, nor developer, nor an organised and cooperating third party. Finally, those such as Cyberguide's IR system are owned and controlled by the developers, built purposefully for specific applications, but rarely see deployment beyond those. We say that each of these technologies constructs a *representation* of location and thus a user's location, or indeed any element of a user's context which can be sensed via various technologies, can have multiple representations.

The issues encountered by the Hitchers team in mapping GSM locations highlight the challenges developers can face when attempting to combine and reason about relationships between competing sensory infrastructures. Each network operator uses a different set of physical broadcast towers and tower IDs so that whilst Drozd was able to develop a quite extensive picture and mapping of the infrastructure provided by two Network Operators, if a user on a third mobile Network wished to use the app, all of that existing data was inapplicable and the mapping process would have to be performed all over again for the new Network. Additionally, users mostly were isolated within their networks so that a user connected to one network could never interact

with a user connected to another because each was located through a separate system. Gathering information about the relationship between the two infrastructures would have required asking users to carry multiple mobile phones connected to different operators, or some manual work of having users label locations and matching up cell tower IDs which share labels. Contemporary to this work, the Placelab initiative [SOHN, GRISWOLD et al.'06] attempted to provide developers with these mappings by encouraging members to war-drive GSM and WI-FI infrastructures with concomitant GPS readings in order to build an accurate picture of the coverage extent of different broadcast towers and wireless hubs. Apart from the massive continual effort required by volunteers to create and maintain data for their locality, this chapter's later section on seams discusses how this data can present its own problems for developers.

The challenges of using GSM as a location technology have been largely ameliorated by the introduction of services for mobile devices such as Android's Location API, which provides a location for the user using either GSM, Wi-Fi or GPS infrastructure and mappings obtained by Google via their own war-driven or crowdsourced information. However, as new location technologies such as powerline positioning [PATEL, TRUONG et al.'06] and piconets [BENNETT, CLARKE et al.'97] emerge, this issue may well rear its head again. Indeed, it persists and has done for quite some time in the offline relationship between the UK's Ordinance Survey and A-Z maps. These are two well established mapping publications that rely on coordinate systems of reference grids distributed across numbered maps & pages respectively. However, being competing map-making companies, the grids used in each system do not correspond and a user of one cannot share grid-references with a user of the other.

Heterogeneity of Location Representations

Locational sensing systems differ not only in their physical infrastructure, but also in how they represent space. GPS provides a Cartesian coordinate system with various other properties that developers may choose to take advantage of, such as user elevation, GSM cell IDs are recorded in Hitchers as the game

progresses and the information used to build a digraph structure representing the cell identities as vertices and their adjacencies as edges so that a user's location can be described as a vertex in the graph. Street addresses, used by many navigation apps, are essentially strings which carry an implicit sense of hierarchy -- House number < Street < Area < Town < Country. Each of these types of representation requires a different reasoning approach to employ it as user context.

Data format is another aspect of variation in representations of context, and one which is closely related to the infrastructure that generates the context data and representation of space that it uses. Systems like GPS and GSM provide location readings that are difficult for human beings to understand but easy for machines to work with, particularly in the case of GPS when determining direct distances. Street addresses on the other hand are universal and human-readable, however they do not naturally lend themselves to operations such as distance calculation. For applications to currently support many representations of a contextual property, the application must be able to parse the formats of each representation and attain some understanding of its structure.

Relating Representations

Whilst understanding a representation's format and structure are often sufficient to reason within that representation, to combine representations a system must be able to reason about further of their properties in relation to each other. For example, representations can also vary in scale. The magnetic field positioning technology employed by [ULLMER and ISHII'97] to track objects used as physical icons (Phicons) in a Mixed-Reality desktop application is limited to a single physical desktop, Patel's powerline system can track the position of users within a building and Wi-Fi technology can provide them with a position across cities, countries and indeed entire continents. To reason across these representations, for example to track the distance travelled when a phicon is moved from its position on a desk to a user's pocket to another desk in a different room, requires the ability to reason about differences in scale between the location systems employed on the desks and across the building.

Often the need to employ multiple representations of a contextual property is driven by the different levels of service and availability that their underlying technologies provide. In addition to covering differing sizes of location area, different applications, technologies and services provide different levels of granularity or precision. This is usually driven by the maximum precision afforded by the underlying technology, however applications can choose to work with sensed data at a higher level of granularity dependent on the services that they provide, for example Groupon [GROUPON UK, '15] - a location-aware couponing app - uses GPS to locate its users, but because the application's aim is to market them a wide range of prepaid vouchers for local offers, this GPS location is resolved only to the accuracy of the city the user is in. High levels of granularity often come at the expense of scale of deployment and vice versa. Whilst ultrasonic location system Bats [WARD, JONES et al.'97] can provide locational accuracy to within a few centimetres, the hardware configuration required limits its deployment to buildings rather than the cities and countries covered by less precise systems such as GPS and GSM.

Newer Forms of Context

Emerging recently are social location systems in which people can label their homes and frequently visited places like work, favourite cafés and friends' houses. Google Now [GOOGLE NOW, '15] makes use of this sort of location data, for example to provide estimates of commuting times. These systems are human-readable and present the opportunity for efficient and highly personalised sense-making since locations that are not significant enough to the user to merit tagging can be discarded. However, they are only subjectively significant -- each user needs their own representation. Future applications of this sort of social location system may involve connecting and reasoning across these different representations. For example: so that "Home" in Bob's representation is understood to be the same location as "Bob's home" in Pete's representation, but only if the system can verify, via a third party social service for example, that Pete knows this particular Bob. In effect the same issues Drozd observed arising from the existence of multiple GSM infrastructures are being translated into the non-physical, social domain.

In the example described above, the label "Bob's Home" isn't sensed, but is input by the user. Zheng's application, which uses amongst other things temperature, weather and crowdedness as elements of user context, is a good demonstration of how context can now be sensed, gathered, or specified manually. In the application temperature is sensed; the weather report is gathered from a webservice; and the user is directly queried by the application as to how busy their surroundings currently are. Burke also uses this direct query approach in a social sensing application and demonstrates how it can be used in seamful situations to verify and correct erroneous data [BURKE, ESTRIN et al.'06] . A context element can also be derived from reasoning upon one or more other context elements. Dey employed widgets and aggregators to achieve this [DEY, ABOWD et al.'01] . These approaches to gathering context further increase potential for the creation of even more representations of context elements.

The advent of social location systems also points to another trend in context-aware computing which is to increasingly employ non-physical context elements. These range from big-data sources where the information is reasonably public, such as social connections on Facebook—these, for example, are used by Highlight to alert users when someone they may know is nearby—to those that involve limited amounts of information that is usually private such as personal calendar events used by services like IFTTT [IFTTT, '15] to control lighting [WITHINGS, '15] and heating [NEST, '15] widgets in the user's home. Applications like these also take advantage of the growing number of smart objects, also known as Things That Think. These objects augmented with sensors can generate user context as data, such as Mediacup's broadcast of temperature and acceleration information [GELLERSEN, SCHMIDT et al.'02] , or in more ephemeral ways such as the Moodfloor [DIX, SHERIDAN et al.'04] , which lights up to show the route a user has taken across the room in a mood-dependent colour, and Fairies, an augmented reality application that adds a CGI fairy to the video feed of a chair every time a user sits in and vacates it [ibid.]

Virtual objects, too, are gaining the ability to sense and react to that which is sensed. Virtual Information Towers [LEONHARDI, KUBACH et al.'99] , for example, can provide information customised to the user who approaches them. And applications such as Stumbleupon [STUMBLEUPON '15] can annotate a website with lists of similar sites generated from tracked user activity.

Context with Dynamic Structure

This generation of a service from the activity of a large group of users can be thought of as another form of crowdsourcing, one that is also becoming more common. Applications like Spotify [SPOTIFY, '16] , last.fm [LAST.FM, '16] and Goodreads [GOODREADS, '16] collect user listening and reading habits in order to build webs of similar artists and authors which can be explored by their users, used by the services themselves to provide users with recommendations for new things to try, or accessed by third parties via APIs. In a context-aware application which accesses these APIs, the value given to the context-element for the user might be a favourite artist or the name of the artist to which they are currently listening.

These sorts of constructed, non-physical context-elements make it plain that some properties of context are *dynamic* or *mutable* in that the underlying set of all possible values of context-element and the relationships between those values undergoes continual change. In the case of a web of music artists, each new band causes a new element to be added to the set of all artists and the similarity connections between bands change over time as listening habits and fashions for genres evolve.

Drozd's work on Hitchers revealed that not only are the structure and properties of non-physical context elements dynamic, the structure of sensed context too can change over time as its underlying infrastructure changes. Firstly, new broadcast towers were occasionally added, on a short- or long-term basis, to the infrastructure of the mobile phone networks they were mapping. This could be seen most clearly around occasional, but regularly scheduled, events that created vastly increased network loads, such as music festivals which are often held in rural locations where call carrying capacity is

low, in these cases additional towers would be erected to increase capacity for the duration of the event and removed at its end. Additionally, the Hitchers researchers were able to observe day-to-day fluctuations in the relationships between cells. For example, the weather could affect how large a cell appeared and thus which cells it was adjacent to.

Discovering Context

Third-party-owned representations such as GSM or Wi-Fi are not only evolving physically, but since our knowledge of them is usually also incomplete, our *models* of these representations evolve as we discover more of their underlying values and structure. Similarly, privacy controls may prevent developers from obtaining unfettered access to non-physical context-element representations such as particular social networks. Unlike static context-element representations, for example GPS, which are bounded and whose structure is known and does not change, we call these representations *partially known*.

Although context may have myriad elements, those elements myriad representations, and those representations myriad models, it is nonetheless possible and indeed common that two applications employ the same model of a particular context-element's representation. In this situation, as Lopes & Fiadeiro put it "this does not mean sensing work has to be replicated [LOPES and FIADEIRO'05] . An appropriate platform of context-aware reasoning would support both the sharing of a single model and its associated streams of data by two separate applications and the combining of multiple models of a context-element so that their associated streams of data might be reasoned across and acted upon by one or more applications. It is the goal of creating such a platform that this work pursues.

Representing and Reasoning About Context

Since our work is concerned with the development of a model and platform for representing and reasoning about context, it is instructive to examine the way in which existing context-aware systems perform these two tasks. The

approaches range from simple if-then operations on text strings to Bayesian logic applied to complex ontologies.

Active Maps – If-Then Reasoning

Schilit's Active Maps Service (AMS) [SCHILIT'95] describes user context through collections of strings structured as key->value pairs. Schilit calls this framework self-describing since, given a suitable method of encoding, any binary data can be entered as a key's value. However, the lack of encapsulated methods in the key->value structure means only a reasoning engine that understands the structure and properties of the data signified by a particular key can use a collection of pairings to reason about a user's context. Locations in Schilit's platform are stored in two formats. The first is as a hyphenated list of identifiers of all locations that contain the indicated location, terminating with the id of the location itself - a sort of reverse street-address format. For example: **JubileeCampus-CSITBuilding-FloorB-Room21**.

Given a set of these strings or the format $l_0\text{-}l_1\text{-}l_2\text{-}...\text{-}l_n$ where each l_i is the id of a location that contains l_{i+1} and is contained by l_{i-1} one can construct equivalence classes $e_0, e_1, ..., e_n$ of these li such that $\forall\ l_i \in e_i$. That is each member of the class e_i is at the i^{th} containment level in the space. In our example **JubileeCampus** would be e_0 the root containment level. The second location format thus can be described as a set of graphs which encode paths between the locations in each equivalence class. Schilit maintains that for speed these paths must be precomputed when the system is initialized which makes it difficult to employ dynamic context-representations. It is also immediately obvious that a location can only be identified and added to the system if its containing locations are known, i.e. the system must possess holistic information about the location before it can utilise it, which might not be possible for a developer working with a partially known representation.

Reasoning in Schilit's platform is then performed in the following ways: To determine which locations contain a user, the AMS can decompose the name of her location into its component parts. In our example above, this yields the relationships:

User is at **JubileeCampus-CSITBuilding-FloorB-Room21**

User in **JubileeCampus**

User is in **CSITBuilding**

User is in **FloorB**

The AMS' reasoning about context is used to infer higher-level information about user state. This is then passed to an application which can reason on it further using If-Then rules, such as IF in Kitchen THEN turn on kettle.

Whilst the graphs provide a way to compute paths between two users, it is much quicker for the system to determine which users are co-located at each level by matching on location ID strings and substrings. This produces relationships of the form $User_1$ is with $User_2$ in **JubileeCampus**. It also illustrates how access to multiple models of a context-representation can be beneficial to developers.

Stick-E Notes – Context Beyond Location

Brown's stick-e note system [BROWN'95] was designed to exploit a Post-It Note information retrieval metaphor with users attaching data or content to a particular context such that they are presented again with that data whenever they enter that context. The basic unit of the system - the Stick-E Note - is structured as follows:

```
<Note [attributes]>

<required>

<with>James<or>Luke

<at>(X,Y)

<content>

     "Remind the guys about our trip to France!"

</note>
```

The required elements: **`with James or Luke`** and **`at (X,Y)`** are computed by a note-triggering module which operates on user context much like the if-then rules of Schilit's AMS. The context that can be described is limited to co-location and position, however Brown argues that states can be treated as positions e.g. **`<at>"Happy"`** or companions e.g. **`<with>"Studying"`** so that information that is relevant to the state can be stored and recalled the next time a user is in the state. Similarly to Schilit, Brown treats position and companions as mere text strings rather than elements of some larger set which may possess a structure relating them so that a user who specifies her state as **`"<at>Joyful"`** will not have the opportunity to access any information stored in notes attached to the state **`"<at>Happy"`**, and perhaps more problematically, a user will not be able to access the content of a note attached to a physical location coordinate unless they are at precisely that coordinate - often a major feat to achieve in GPS positioning. Flintham et al's experience with the Bystander application [FLINTHAM, BENFORD et al.'03] demonstrates the difficulty systems can encounter here with online coordinators, who are responsible for triggering content delivery when their situated partners reach a certain GPS position, having to make a number of compromises due to the inaccuracy of the GPS data, from assuming their partner's position based on what they describe, to triggering a piece of content anyway when it becomes clear the situated user is struggling to attain the correct GPS position.

CoBrA – Agents and Ontology

The CoBrA system [CHEN, FININ et al.'03] is designed to support context aware intelligent spaces. The system employs an agent architecture to collect context and a common context ontology through which agents share information with a central context-broker. The ontology defines the types of places, agents, locations, and activities that the system reasons over and their properties such as the coordinates of a located agent or user, or the start-time of an activity.

Although this ontological approach supports extremely expressive reasoning, the system is difficult to extend on the fly since each element of the ontology must have a defined relationship to the others to enable the Broker to reason about it. Similarly to Chen, Shehzad's CAMUS provides an ontology of the home domain for supporting smart homes, which illustrates that (as proposed by Dix [DIX, KATIFORI et al.'10]) the use of ontologies is best confined to smaller, well-defined domains, although strategies can be employed to train a system on an unfamiliar ontology [DIX, KATIFORI et al.'10] .

CLAR – Fuzzy Context

Zheng's CLAR [ZHENG, ZHENG et al.'10] , a collaborative activity recommendation system employs a location coordinate system based on GPS which encodes context as stay-points and stay-regions rather than directly using the location coordinates given by the GPS sensor as the user's context. A stay-point is a set of points in a user trajectory whose distance from each other is very small relative to the time between them being registered. They consist of an average x and y coordinate and arrival and leaving times. Stay regions are constructed by applying a clustering algorithm to the set of stay points and in this way users and activities can be assigned their location based on the region the system reports them to be inside.

Information about the city is recorded in two matrices, one storing relationships between locations, attractions, and activities, and another which ranks the similarities between two activities. When a user is detected at a particular location, these matrices are used to look-up activities to suggest to them. The system constructs these matrices and the stay-region locations prior to runtime using other applications. Although it is possible the matrix population algorithms could be adjusted to be applied at runtime the specialisation of the matrices means that new aspects of context cannot be added to the system without significant effort on the developers' part.

Planning with Matrices

Vukovic's context-aware platform [VUKOVIC and ROBINSON'07] also makes use of matrices. These are combined with vectors into a structure he refers to as

a “context mesh”. This is utilised alongside a goal taxonomy as the informational foundation of a context-aware planning engine which delivers customised information and behaviour by chaining context-appropriate services to satisfy user requests. For example, to satisfy a request for a restaurant recommendation by an English-speaking user driving through Zurich, the planner chains RestaurantFinder, DirectionsFinder, Translation and Speech services according to relationships expressed in the goal taxonomy and context mesh. This provides a powerful and expressive context-aware planning system, but one which requires a comprehensive description of its operating domains to be constructed and finalised before runtime. Gross' vector representations [GROSS and PRINZ'03] employ a similar approach to Vukovic.

Self-Organising Maps – Clustering Context representation sets

At the heart of Vukovic's planning engine is an AI planner, which determines the correct chain of services that will move the system through various states restricted by pre- and post-conditions which are defined by the user's context. Vukovic’s approach requires a description of the operating domain's structure, however other work [CAKMAKCI, COUTAZ et al.'02, VAN LAERHOVEN and CAKMAKCI'00] with context as sensed by wearable smart garments uses Self-Organising Maps (SOMs) and Hidden Markov Models (HMMs) which can reduce the amount of developer or user involvement required in modelling various properties of user context. A Markov model is one in which the value of the next output is obtained from some manipulation of the current output. In a HMM only the stream of outputs is known whilst the manipulations that produce one from the preceding remain obscured. The utility of this construction for a context reasoning platform can be seen if we return again to the Hitchers application, which locates users within GSM cell IDs that can also contain virtual objects called Hitchers, which users may search for at their GSM cell ID location, pick up, and carry around with them, before setting them down again in another GSM cell. The app’s location model was formed by recording the sequence of broadcast cells that users passed through during the course of the game and using these sequences to construct an adjacency graph of all observed GSM cells. When a new cell was observed, all that was

required to connect it to the graph was the id of the last observed cell. In this way, the Hitchers system was extremely tolerant of the partially known context representations [DROZD, BENFORD et al.'06] . Similarly, a model that only requires the system understand the previously observed state to construct and reason on the currently observed state supports any partially known context representations constructed in this manner of sequential discovery.

SOMs can be used in context-aware applications to cluster sensor data in a spatially structured manner. A SOM employs a set of neurons arranged in an n-dimensional (usually n=2) lattice where each neuron has a vector of weights of equal dimensionality to the input the SOM will receive. The neurons are trained in such a way that those close to each other in the lattice become excited by similar input vectors. This approach can construct a topological representation of non-spatial and high-dimensional context data which categorises similar types of input as spatially near to each other in the neuron lattice [VAN HULLE'12] .

Intelligent Classroom – Layering Behaviour

Franklin's work on intelligent environments [FRANKLIN and FLASCHBART'98] also employs planning and takes a robotics inspired approach to context, considering the interior of a smart classroom to be a world in the way that a robot would consider the things its sensors detect external to it to be its world. As with a robot's behavioural architecture, reasoning is performed in several layers which employ tight control loops mediated by the room's sensors. The classroom's behaviours are split into discrete skills which are activated and parameterised as needed by a contextually driven execution system. Combined with a planning approach to reasoning that includes building a plan of the user's behaviour, this allows the classroom to synchronise its behaviour with that of a human speaker. This approach is powerful, but again is limited in the scale of its distribution as there is no provision for entities in different rooms to interact with other rooms or with each other.

Spreading Activation – A Cognitive approach to context

Dix's work in [DIX, KATIFORI et al.'10] takes a cognitive approach to context, constructing a personal ontology for the user and reasoning contextually upon it using spreading activation. The driving principle of this is that information recently accessed, or information similar to information recently accessed is most likely to be the information required next. And so, for example, if a user is looking at the Facebook page of a friend with an upcoming birthday in their calendar, the date of the birthday and other information about the friend such as their address will be activated. If the user then visits a flower delivery website, the date of the friend's birthday, being active, could be automatically suggested as the flower delivery date. It is easy to see how this approach could translate to physical space with a user's path through the space creating a fading trail which engages any sources of information within its locus with probability of engagement increasing proportionally to how recently the source was passed. This temporally based approach to context is employed in a more ephemeral manner by moodfloor[DIX, SHERIDAN et al.'04] , an installation that tracks and displays users' footprints across a floor on their way to a coffee machine. At the machine users are given the opportunity to specify their mood, positive or negative and their footprints change colour according to this choice. Over time image of the footprints fades so that the department is treated to a dynamic piece of art and a continuously updated visual snapshot of the current moods of its faculty.

Similarly, to Cakmacki's SOMs, the spreading activation system must first train upon the ontology, weighting its entries in a process that can be thought of a pseudo-clustering of the ontological entities. This implies again that the domain must be known in its entirety before runtime, although the work asserts that weights can be adjusted once the program is running.

Dix's algorithms differentiate between information that is important in the short- medium- and long-term. This distinction allows some pieces of information, e.g. a user's home telephone number and address, to carry more general significance in the system than others, physically this is analogous to a the scale of an object and a sense of nearness with respect to that scale. This is

both a distinction and an abstraction that can be lost in other systems where the focus is either purely on the binary of co-location or the absolute of distance without scale.

CALAIS - Geometric approaches to Context

Computationally geometric approaches to representing context tend to be confined to reasoning about physical location. Harter [HARTER, HOPPER et al.'02] and Nelson [NELSON'98] apply geometric approaches to their location-aware applications, storing user and object context (location) in quadtree structures. This provides very efficient ways of inserting, removing, and finding entities. Harter also encodes orientation by designating some faces of an object's bounding box as active. i.e. interaction can occur at this face; for a user this would be the front of their body. The active face is stored as a pair of coordinates with the coordinates of the opposite face stored to indicate direction (an active face can only look out of an object, not into or through it.) Interaction between two entities in Harter's model is thus only possible when they are close to each other and their active faces are oriented appropriately. Although it is extremely efficient, it may be difficult to apply this geometric approach to non-physical context such as social connections.

Storytelling – Seating Context in Physicality

Some applications allow the physicality of context to take care of the reasoning. Fails' storytelling application [FAILS'09] uses the adjacency of sensors attached to mobile devices to trigger screensharing behaviours. This sort of ad-hoc and intuitive approach to user context may lack the expressive power and flexibility of other less lightweight approaches, but it captures what McCullough, echoed by Crabtree & Rodden [CRABTREE and RODDEN'08] identifies as an essential aspect of mobile ubiquity in urban spaces: the "physical scope" of connectivity, which supports a paradigm that is both intermittent and local.

Context Toolkit – An alternative approach to context

Some context-aware systems are less concerned with the details of representing and reasoning about context and more with providing an extensible platform to

allow developers to create their own representations and reasoning modules; chief among these is Dey's Context Toolkit[DEY'00] . The toolkit provides collections of context-widgets, which abstract context, hiding the details of sensor implementations and allowing data streams to be reused; interpretation-widgets, which provide mapping from lower level machine-readable contexts to those that are more abstract e.g. from GPS to street address; aggregation-widgets that collect similar context data streams together; and actuation and output services which provide data and behaviours to applications. A collection of discoverers keep track of these so applications don't require the network addresses of components. Although it does not provide a framework for reasoning about and across differing types of context, once a widget has been constructed to map from a less abstract to a more abstract context, it can be reused indefinitely and therefore lessens the complexity of developing future applications.

Space

The model and platform presented in this work take a spatial approach to representing and reasoning about context. Spatial reasoning is deployed in various manners in a variety of fields, including Computational Geometry, Robotics, GIS, Spatial Economics, Architecture and Computer Supported Cooperative Work. In this section we examine some of the ways in which each of these fields treats spatial reasoning and how their methods and approaches can inform our own spatial model of context. We conclude the section with an overview of Benford's CSCW Awareness model and Dix et al's presence and performance taxonomies[DIX, RODDEN et al.'00, DIX, SHERIDAN et al.'06] , upon both of which our model and platform based.

Modelling Space in the CALAIS System

First, however, we consider Nelson's [NELSON'98] work, discussed in the previous section, and being one of the earliest Pervasive Computing approaches that treats space directly, in more detail. The application - CALAIS -- approaches the issue of modelling both the position of users and objects in space and their orientation. In order to achieve this the system represents entities as rectangles with one or more sides designated as active, through

which interaction can take place. When two entities' active edges are facing and within an appropriate distance of each other, the system will initiate interaction between them. Each of these edges are maintained in the system as the coordinates of a pair of line segments representing the active edge and the edge directly opposite (behind) it. The opposite edge is stored to allow the appropriate side of the active edge to be determined so that active edges face outwards, not into, an entity's "body".

Nelson maintains two spatial models in his system using both Cartesian coordinates and, borrowed from Computational Geometry, an R-Tree structure. The R-Tree successively divides the space to be modelled into Minimum Bounding Rectangles (MBRs) with the node at the root of each subtree being the MBR for the set of rectangles in its subtree. Harter and Hopper [HARTER and HOPPER'94] —who propose a similar 2-representation approach, modelling orientation in their BATs system using the position of 3 non-co-linear points on the entity—used the more traditional version of this approach, the quadtree, where areas are successively divided into quadrants.

Modelling Geometric Space

Computational Geometry tends to take three approaches to modelling space. These are grids and trees, which are populated with a variety of geometric primitives; and linear or quadratic equations which describe objects as the intersection of several half-planes. Whilst in the context-aware approaches of Nelson and Harter et. al., trees are used to facilitate searching, in graphics engines such as those used for console games, grid and tree approaches are designed to allow the system to disregard as much of the data as possible, focusing computation on the areas within which intersections are likely to take place given the movement of a particular object. This approach reduces the number of comparative operations that the system must perform.

Grids can be uniform, in which case the system is split equally into a number of smaller squares or cubes, and objects are referenced by every square that they overlap. This approach, however, cannot be applied to domains whose set of squares could be infinite or at least very large. Instead developers can use a

hashed-grid which stores only the coordinates of squares that are populated. We can see the potential application of this for a sensed context representation which is being explored by the system at runtime. Although the extent of the context representation may be much larger, only a portion of this is made available to the system to be reasoned upon at any one time. A variation of the Grid structure: Hierarchical grids, uses a grids of a tree structure, however they do not require a single root node and thus can begin with a size and distribution of partitions appropriate to the size of their largest elements. This approach saves both space and node traversals.

Teller's cells and portals approach [TELLER'92] similarly divides space into regions traversed via portals. The method was conceived to support architectural walkthroughs and prioritises accurately portraying the isovist—the volume of space that can be seen by a user from a particular viewpoint given the structure of the building or environment around them. A cell represents a single room with one of more portals, represented by bounded hyperplanes[2], indicating doors or windows from that room into other spaces. Objects are only checked for collision with other objects in the same cell or whose cells they can directly access via portals in their cell and adjacent cells. As the divisions are made based on the structure of a (notional or not) physical space, this approach to spatial modelling could be useful for context-elements whose representation is a Cartesian set with some form of structure.

Clustering Trees

One form of computationally geometric tree that is interesting to us is Garcia's n-ary clustering tree [GARCÍA, SAPPA et al.'99] . Given a set of spheres, Garcia defines the attraction between any pair as: a $= \frac{r_1 r_2}{d}$ where r_1, r_2 are the radii of the spheres and d is the distance between them. Thus attraction is proportional to the size of the spheres and inversely proportional to their distance.

The attraction between two spheres is used to weight the edge between them in an adjacency graph and this graph is used to construct a Minimum Spanning

[2] objects of one less dimension than the space they are embedded in

Tree (MST) of the spheres. The edges of this MST are ordered in ascending weight and nodes of the edge with the lowest weight are combined into a bounding volume, or cluster. This process continues with all proceeding edges with the caveat that if a node is to be grouped with one that is already a member of a cluster, it is instead grouped with that entire cluster. This process, which takes $O(n \log n)$ time, produces a Binary Clustering Tree that exhibits a natural clustering effect.

As clusters are constructed, they are assigned a grouping cost based on the weight of edges involved in their construction. Garcia's final step transforms the Binary Clustering tree into an n-ary clustering tree by merging nodes of similar grouping cost. Graphics systems can also combine spatial representation by embedding tree structures within grids, or using grids to provide quick access to subtrees of the main tree [ERICSON'04] . Later in this Chapter we see a similar use of embedding with Hierarchical Voronoi Graphs.

Modelling Space with Computer Graphics Techniques

Mapping the space itself to a structure is only one part of the process required for spatial reasoning. We must also map the objects that populate the space onto some sort of useful representation. In graphics, the most common approach is to reduce objects to more simple approximations of their shape. This can be done by enclosing the object in the smallest primitive that will completely contain it. Since these primitives are always equal to or larger than the object they represent, systems need only test two enclosed objects for intersection if their primitive enclosures collide. This abstraction gives the system the leeway to maintain underlying complex representations of objects since these are now accessed less frequently.

Some types of primitive employed by graphics systems are:

- Sphere or circle
- Axis-Aligned Bounding Box (AABB) - this is the smallest box with edges parallel to the space's axes that completely contains the object. Since the edges of the box can be projected onto the axes, detecting overlap between two boxes in an n-dimensional space is a matter of

detecting overlap between their N pairs of intervals, projected onto the axes.

- Oriented Bounding Box (OBB) - similar to an AABB, but with the requirement for parallelism with the space's axes removed.
- K-Direction Discrete Orientation Polytope (K-DOP) - a shape constructed from K/2 pairs of parallel half-planes whose orientation is fixed across every object. Since orientation is fixed, its overlap test has similar complexity and expense to the AABB test.
- Convex Hull - the smallest convex polygon equal-to or containing the object.

For the context-aware platform employing these primitives in a Cartesian space, various well-tested algorithms exist to determine intersection between them. The interested reader can consult Ericson [ERICSON'04] and [CHEN'96] for further details

Polytopes described by arbitrarily oriented half-planes are also easily described as a set of linear inequalities and the intersection of a pair of these polytopes can be restated as whether there exists a solution to the union of their inequalities. Again, there are many algorithms that can solve this sort of problem, however, if as is often the case in context-aware computing, we are interested in the distance between 2 polytopes, or the depth of their intersection, we must employ quadratic inequalities and there exist few if any efficient algorithms for finding solutions to sets of these.

Abstracting Geometric Models with Voronoi Graphs

Robotics and, in particular, robot navigation is another field which makes use of concepts from computational geometry. Robot navigation typically takes either a geometric or a symbolic approach to space although hybrid approaches are also used [AFYOUNI, CYRIL et al.'12] . The geometric approaches map the space using the sorts of trees or grids previously discussed, or decompose it into Voronoi Graphs.

Given a set S of points in $\mathbb{R}^n$, $d: S \times S \rightarrow \mathbb{R}$ - a distance function on S & min a mapping with the properties:

$$min: \mathbb{R}^n \to 2^S$$
$$\min(p) = \{s \mid \forall\, r \in S\,.\; d(p,s) \leq d(p,r)\}$$

i.e. $\min(p)$ returns the set of one or more members of S minimally distant from p.

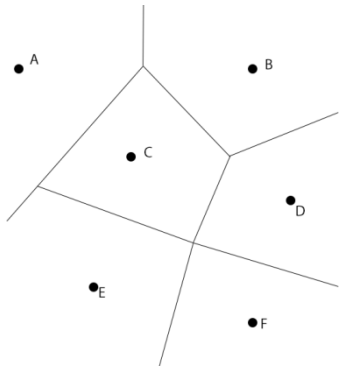

Figure 1 - Partitioning space with min function

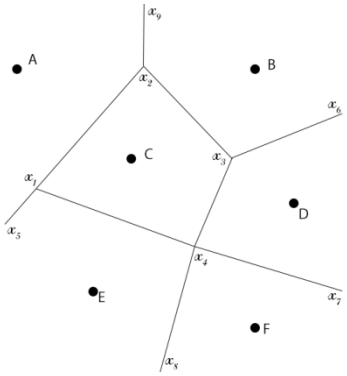

Figure 2 - Meetpoints and endpoints of partitioning line segments are identified

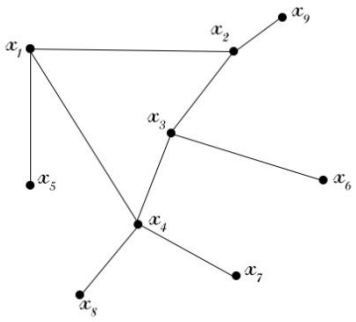

Figure 3 - Voronoi Graph is constructed from labelled meetpoints and endpoints

The Voronoi Diagram VD_s of S is then the set of all line segments and half-lines in $\mathbb{R}^n$ formed of points with exactly two minimally distant points in S and meeting at *meetpoints* with at least three or more minimally distant points in S. Figure 1 shows the way in which the set S = {A, B, C, D, E, F}, and the mapping *min*, partition a space into areas of points minimally distant from some point in S. The lines are composed of points that are minimally distant from more than one point in S. The points where the lines meet are points that are minimally distant from more than two points in S. If the points in S represent obstacles, the lines can be taken as unobstructed paths through the space. Figure 2 shows the labelling of these meetpoints as x_1, x_2, x_3, and x_4. These are combined with the endpoints x_5, x_6, x_7, x_8, x_9, to create a Voronoi graph that describes in abstract the structure of paths through the space (Figure 3). A Generalised Voronoi Diagram (GVD) is one whose set of points, S, represents the hull of a collection of arbitrary geometric objects and thus consists of a collection of points in $\mathbb{R}^n$ that have at least 2 minimally distant points belonging to different objects in G. Instead of straight lines and line segments, a GVD consists of curves. Some curves have *endpoints* on the objects themselves where concavity exists. Generation takes typically $O(n \log n)$ time [WALLGRÜN'09] .

A Generalised Voronoi Graph (GVG) is an undirected pseudograph (one that may contain loops and parallel edges) containing *Voronoi nodes* for each meetpoint and endpoint in the GVD and an edge between each pair of nodes wherever a curve exists between their represented meet- or end-points in corresponding the GVD. The GVG can be said to provide a representation of the GVD that is absent such geometric information as distance and direction.

A combinatorial embedding into a GVG creates an Embedded Generalised Voronoi Graph (EGVG). This provides a counter-clockwise ordering of edges at each vertex that corresponds to the ordering of curves entering the corresponding meetpoint or endpoint in the original Voronoi Diagram. Robotic navigation systems also make use of Annotated Generalised Voronoi Graphs (AGVGs) which attach information such as environmental descriptions and GPS estimates to vertices and attach length, minimal clearance values, and

exploration status to edges of the graph. When applying these processes to an arbitrary space populated with many obstacles and walls, this yields an efficient mapping of paths that a robot may take through the space.

Wallgrün presents an extension to this: the Hierarchical Annotated Generalised Voronoi Graph (HAGVG) [WALLGRÜN'09] which maps, and supports navigation through, spaces which are only partially known and may be dynamic. At the bottom of the Hierarchy, the most detailed level, is an AGVG which represents what the robot knows or thinks it knows about the space. Subsequent levels consist of graphs which abstract the graph on the level below by collapsing one or more of its subgraphs into a single node or edge.

The relevance of a node in a graph is determined by the number of regions accessible from the edges that leave it, the size of those regions and their secludedness from the rest of the graph. A subgraph may be collapsed to its single most relevant node; or if its paths constitute a loop from a relevant node, or a set of paths between two relevant nodes, it may be collapsed to a single edge. This structure supports more efficient navigation by allowing reasoning about the space to take place on a level, or levels, of abstraction appropriate to the current navigational task and regardless of the robot's certainty about less significant details of the space's layout. For example, plotting a route from one side of the building to another can be performed at a level of abstraction which hides the navigational details of alcoves and rooms with only one entrance or exit and allows choices such as "Should I follow the left balcony overlooking the lobby, or follow the right balcony overlooking the lobby?" to be deferred to a later point in the navigation process.

Not only do we find this sort of spatial structuring and abstraction to be interesting for context-aware goal attainment, but the abstraction processes that Wallgrün uses could support abstraction modelling of graph-based context-element representations that increase efficiency in the same way that bounding polygons and grids and trees can do for Cartesian representations. Further, since the system is conceived to be used by a robot which is exploring an unknown and dynamic environment, such graphs in a context-aware system

can conceivably be built at runtime and updated incrementally by a server as information about the context-representation trickles in.

The other approaches to navigation frequently employed in robotics are classified by Afayouni et al. as symbolic. These are approaches to robotic navigation that use human-readable descriptions and support topological relations via sets and graphs. Whereas geometric representations tend to focus on space and paths through it, symbolic approaches can be said to be more concerned with place—that is they identify various subspaces of a larger space by their function rather than their geometric properties alone. Because the symbols employed in such systems carry both structural and functional meaning, they can be reasoned about both spatially and semantically, for example a geometric context model would return the nearest canteen in the building as a result for the query "find the nearest place to eat my packed lunch", but a symbolic model which connects locations of similar function might return results such as canteen, roof terrace garden, and staff break room.

It can be useful, for example when providing services over large buildings, cities or countries, to embed locally relevant symbolic models in larger geometric spaces. The hybrid models of space employed by robotics researchers surveyed in Afayouni et al offer either a patchwork approach that stitches a set of local models together into a spatial quilt, or parallel symbolic and geometric models that overlaying each other describe the same space. However, they stop short of providing the sort of abstraction relationships that Wallgrün's HAGVGs do. And, perhaps due to the difference of goals between robotic navigation and context-aware reasoning, we do not find any exploration of the processes of generation and reasoning about more complex and flexible approaches. For example: partially overlapping one mapping of a space with another; embedding one mapping of a space into gaps in another; and any combination of multiple of these embeddings, overlaps, overlays or quilts.

Space in Architecture, Town Planning and GIS

Architecture & town planning and GIS are two more fields that concern themselves with the modelling of space. Like Hierarchical Voronoi Graphs,

GIS provides a layered representation of space, but as the goal here is often data-visualisation, filtering and manipulation. Each layer represents a different set of information about the space in question. For example: a system might allow queries to produce the simultaneous visualisation of statistical data, such as life expectancy, average salary, and pollution levels, across a city so that inferences can be drawn about the impact of socioeconomic factors on longevity.

Architecture and planning, on the other hand focus on the structural qualities of space. Whilst the structure of interior spaces - of rooms, corridors, stairways and the means of moving between them - can be easily represented as graphs, exterior spaces, particularly those in urban environments, appear more complex. Rather than subdividing space into smaller, well-defined subspaces which can be traversed via also well-defined doorways, the structures of exterior spaces seem to grow out of them, populated by often irregular (especially in older European towns) groupings of buildings, and composed of streets, squares, courtyards and junctions that flow into each other so that it can be difficult to delineate where one ends and the other begins, exterior space can appear as though it lacks any regular, definable structure at all.

Hillier provides a method for decomposing this seemingly unstructured space into axial and convex graphs [HILLIER and HANSON'89] which model those parts of the space respectively defined by their length and their breadth. Again, we see that it is often useful, nay necessary, for space to be described in multiple ways. Hillier notes that these two classes of space often comprise different functions and attract different populations of the city. For example, strangers to an area frequently have more connection to axial spaces which provide them with access through the system, whereas convex spaces create static zones where those more familiar with the local area enjoy more control and oversight with respect to access to its private interior spaces. This difference implies that not only can models of space differ according to the technologies that generate them, or according to implementation choices imposed upon them by design requirements such as precision and efficiency, but developers might also wish to consider sociological aspects of use. That

social aspect to space Hillier includes, echoes McCullough's insistence that pervasive environments differ fundamentally from those of cyberspace in that they must be "inscribed onto both the structural and social features of the existing environment"[McCULLOUGH'04] Considering this work from a ubiquitous computing point of view raises the question of whether it may be possible to similarly classify areas of cyberspace by approaching the division as one of use and behaviour rather than of shape.

Another way to approach Hillier's model is as a separation of space into areas traversed quickly and those lingered in. From an experiential point of view, although a city square and a junction take up the same amount of space, the square, being part of the convex graph, occupies a larger significance to the residential user who takes lunch and spends evenings socialising in the cafes that border it, than the axial-situated junction which is quickly traversed on the way to other destinations. Further to these considerations, cognitive science tells us that people tend to create mental models of their environment which are both fragmented and distorted by the significance with which they view particular environmental features [TVERSKY'92] . Chalmers'[CHALMERS and GALANI'04] and Crabtree's [CRABTREE and RODDEN'08] work in seamful applications demonstrates that this is also the case in Mixed-Reality environments where the sense of space and the cognitive model straddles the virtual, physical, and augmented worlds. These properties of these hybrid spaces - which Crabtree et al. refer to as hybrid ecologies of space have implications for context-aware applications that employ location, since for example: a navigation application may wish to emphasise axial spaces over convex and vice versa with a recommendation application for tourists which seeks to provide them a more authentic experience of the city.

Further to this, when Pervasive technologies can be seen to distort, bring forward or cause breakdown in aspects of these hybrid spaces [FLINTHAM, BENFORD et al.'03] , the role of the user's cognitive model of the space can become even more important. Designing Pervasive systems in order to exploit such distortions and breakdowns is referred to as seamful design, and in the section on seams and in further chapters, we discuss this aspect of Mixed-

Reality and Context-Aware systems, as well as approaches more generally mindful of seams.

Deforming Space for Fun and for Profit

The deformation of space is not limited to cognitive models. For instance, the stretching and compression of space is employed deliberately in the field of Spatial Economics where distance between two points is expressed in non-spatial values, e.g. the cost of transporting grain per ton, per mile. This non-spatial data, encoded into a cost function, is then used to produce a visualisation that distorts the original space according to the cost function, so that for example: a map of Britain appears pinched around its major motorways and railways, with areas like Norfolk, where good road and rail links are less prevalent, taking on the appearance of being much larger than they in reality are; a phenomenon surely familiar to anyone who has found himself stuck in roadworks on the A12 and a situation which neatly illustrates the utility of spatial distortion in a context-reasoning system since in some situations projected travel time can be a key consideration; for example, in a restaurant recommendation system. Distortion of the space can also provide an effective structure in otherwise unstructured Cartesian spaces. McCullough gives the theoretical example of a pervasive printing service that sends an office worker's document to a printer owned by a different firm in the same building because his office is located next door to their printing room making their printer physically nearer to his location than any operated by his own company [McCullough'04] . A deformation of the spatial model maintained by the printing service, which creates an artificially large separation between the two firms' offices, would correct this issue. This structuring of space according to how far, and whether or not it is possible, to reach one location from another also arises in the natural world with rivers and chasms, which cannot be crossed directly but must be travelled along until a bridge or fording point is reached, and elsewhere in the built environment with one-way streets, which can disconnect the distance from one location to another from their as-the-crow flies distance.

Presence and Awareness

Our approach to context representation and reasoning is informed through considerations raised by all of the previously discussed work, but it builds directly on the collective work of Benford, Dix, and Rodden in CSCW. We take their approach to user presence in virtual reality applications and apply it to a range of contexts, both spatial and non-spatial as part of a flexible hybrid model that allows different context-representations to be combined and reasoned about as any number of embeddings, layers, quilts and overlays.

Benford and Fahlén [BENFORD and FAHLÉN'93] describe a system for managing user interactions in Large Scale Virtual Environments (LSVEs) that is designed to limit the amount of processing the system must do in order to determine which audio and video streams should be directed to each user. For example, at a virtual networking event where users are represented as avatars moving around a virtual room, where those avatars' faces are imposed with video feeds and conversations over audio and video can be initiated between avatars. For the system to determine in an isovistic manner which video feeds a user can see and how those feeds appear to them, and additionally which audio feeds to transmit to them and how these should be combined, would be extremely expensive. Benford instead proposes each user's avatar be endowed with several *Aura* - defined areas around the avatar that indicates the extent of the user's various areas of interest and influence - respectively, their *Focus* and *Nimbus*.

When two users' aura intersect, the system calculates how they should experience their interaction, for example, the gain level to which an audio feed transmitted should be set, or what level of compression to apply to a transmitted video feed. When two users' aura intersect, Benford says that they have *awareness* of each other. This limiting of computation regarding the details of interaction to proximate concerns alone has, in a similar manner to bounding primitive and tree-based approaches in computational geometry, the effect of markedly reducing the complexity of the system, allowing environments that support greater numbers of users to be built. The system also

benefits from a measure of intuitive usability: users can see at a glance what their awareness is and easily understand how various actions might affect it.

In Mixed-Reality systems, the areas of the system that a user can affect and those points within the system from which they can observe others are collectively referred to as their *presence*. These map respectively to their Nimbuses and Focuses in the Awareness model.

Dix [DIX, SHERIDAN et al.'06] presents a continuous taxonomy of performative presence in Mixed-Reality systems which can be decomposed into four different quadrants these are detailed in Table 11

	Public action	**Private action**
Public results	Neutral performance	Magic
Private results	Suspenseful performance	Private interaction

Table 1 – Classification of Performance vs. Privacy

An adaptation of this taxonomy forms part of our approach to user awareness within our own awareness model.

Extensions to the presence and Awareness Archetype

Regarding context, is it mentioned in [PEDERSEN and SOKOLER'97] that ubiquitous systems must strive to strike an attentional balance between support and distraction and the Awareness model achieves this elegantly. The system has been extended with additions such as 3rd Party Objects which allow users to alter their aura in intuitive ways, such as picking up a virtual microphone to increase the size of their audio nimbus. Conceptual extensions of the Awareness system were proposed and explored by Dix [DIX, RODDEN et al.'00] and Rodden [RODDEN'96] . Dix imagines an awareness model that supports both Cartesian and topological (graph- or set-based spaces) and, drawing upon the concept of manifolds, allows smooth transitions between them. Rodden

presents a generalisation of a single-space awareness model to support topological, networked and non-spatial domains, with two separate aspects of Awareness: Continuous and Discrete.

Rodden's model presents user presence as multiple ordered pairs which consist of a position point and some subset of the space, which he refers to as the position point's "*Adjacent*" points. Since the model permits non-spatial sets there is no requirement for members of these subsets to follow a geometric definition of adjacency, additionally a user can have multiple position points within a single set. The model provides functions for Nimbus and Focus which map a user and one of the unique ID's of their set of Aura to a set of presence pairings in the space. Functions for aggregate location, focus and nimbus return the full set of points in the space at which the user is located or which are covered by each of their Aura. Awareness between users in Rodden's system [RODDEN'96] is provided both by an awareness strength function, which maps pairing of users to a numerical awareness value based on the overlap between their aggregated focus and nimbus, and by an awareness mode function, which Rodden describes as mapping a pair of users to one of either ten Aura-Position configurations or six Aura overlaps (some combinations are discarded as semantically equivalent) that are described below.

- **Mode 1** complete disconnection
- **Mode 2** B contained within A's Nimbus
- **Mode 3** A contained within B's nimbus, B contained within A's nimbus
- **Mode 4** B contained within A's Focus
- **Mode 5** B contained within A's Focus and Nimbus
- **Mode 6** B contained within A's Nimbus, A contained within B's focus
- **Mode 7** B contained within A's Focus and Nimbus, A contained within B's Nimbus
- **Mode 8** A contained within B's focus, B contained within A's focus
- **Mode 9** A contained within B's focus, B contained within A's focus and nimbus

- **Mode 10** A contained within B's focus and nimbus, B contained within A's focus and nimbus

- **Overlap 1** A and B's nimbus overlap
- **Overlap 2** A and B's nimbus overlap, A's focus overlaps B's nimbus
- **Overlap 3** A and B's nimbus and focus overlap
- **Overlap 4** A and B's focus overlap
- **Overlap 5** A's focus overlaps B's nimbus
- **Overlap 6** A's focus overlaps B's focus, A's nimbus overlaps B's nimbus.

Table 2 - Set of modes and overlaps of user Aura from [RODDEN'96]

Rodden also proposes a `field_awareness()` function that can describe strength of awareness over the overlap between a focus and nimbus whose respective strength at any point in the set is described by the functions `focus()` and `nimbus()`.

The paradigms of presence and Awareness fit neatly into Dourish's interpretation of context as something that is continually negotiated between users, since where a user directs her aura and how large she makes them directly affect what interactions she experiences and partially, but not fully, control how she perceives and is in turn perceived within the system. The other "part" of the user's perception and appearance is driven by the influence of other users' aura that intersect with her own. Through the awareness that arises between users when their Aura intersect, the system mediates user interaction.

No surprise then that this approach has been applied to location and context-aware systems previously. Hitchers uses focus to determine what users see when they search a location. Recalling that users are located through mobile GSM broadcast cells in a digraph of adjacent broadcast cells, the size of the user's focus dictates how many hops are taken when searching the local graph for virtual creates (created and deposited in cells by other users) to display. Other work, [MAISONNASSE, GOURIER et al.'06] , approaches the paradigm from a mechanics standpoint, positing Aura as similar to gravitational fields,

and applying Newton's first law to two proximate users to determine if they are engaged in shared activities for which the system should then attempt to provide support.

Further additions to the paradigm in CSCW also inform our model. MASSIVE: the Model, Architecture and System for Spatial Interaction in Virtual Environments [GREENHALGH and BENFORD'95] implements the Awareness system at the core of its spatial model. A third iteration of the architecture: MASSIVE3, draws on SPLINE's paradigm of [PURBRICK and GREENHALGH'00] to support multiple spaces within the system. Each locale is an encapsulated space, distinct from the others with its own coordinate system. Movement between locales is by way of Boundaries which are defined by:

- A target locale
- A transform which places objects in the target coordinate system
- A polygonal area that users move through to exist from their locale and enter the target
- And an effect of the boundary on Awareness

Like Dix's manifold space paradigm, this approach allows structurally different spaces to co-exist in the same system. In our model, users may have presence not just in one space, but in multiple structurally diverse spaces at once. Some of these presences will be generated by the spaces a user inhabits and others by the relationship between these inhabited spaces and spaces they are connected to. We adapt MASSIVE3's Boundary system to achieve this projection of users' presences into spaces that they do not natively inhabit.

Qualitative Spatial Reasoning

Once two users' Aura collide in the Awareness model, the system begins to calculate the extent of the overlap and how that should affect the users' interaction. As in Rodden's generalised model of Awareness, in context-aware applications, particularly in topological or symbolic systems of representation, it is useful to be able to reason about how entities are arranged in a space, or the general ways in which their states relate to each other, rather than the

precise value differences between those states. Such reasoning is known as *Qualitative*.

One of the main motivations for qualitative spatial representation and reasoning according to Renz [RENZ'02] is its similarity to human cognition, and one of the most frequent criteria for qualitative reasoning systems, Renz goes on to point out, is their "cognitive-adequacy" or "-validity". In informal terms this refers to the similarity of the reasoning system's approach to the way most people might think about and categorise situations. The symbolic robot navigation systems surveyed by Afyouni are also qualitative and since they rely on human-readable instructions such as "turn right at the fountain".

Cognitive-Adequacy can be broken down into Conceptual-Adequacy, referring to adequacy of the model's vocabulary; and Inferential-Adequacy, which refers to that of the reasoning process. Renz's experimental work can be used to define conceptual-adequacy more formally for spatial reasoning, as a continuum rather than a binary property, in a manner which can be summed up as follows: Assume that to group a set of images by similarity without bias is to be presented with a set of unlabelled images and asked to group the similar images together without being supplied with any information that could lead or direct grouping such as the number or names of categories of similarity. Where subjects are given any set of images depicting, in a number of different ways, each of the spatial configurations employed in the Qualitative reasoning system, and directed without bias to group those images according to their similarity, a QSR model's cognitive-adequacy can then be described as the proportion of users who sort the images into a number of groups equal to the model's categories and with each group matching exactly the content of its associated model category.

In [RANDELL, CUI et al.'92] the authors present RCC-8 a Region Connection Calculus which defines a complete Boolean lattice of 8 types of possible spatial configuration between two objects in a plane. These configurations are given in Table 322.

$\boldsymbol{PO(a,b)}$	partial overlap (the interiors of sets a and b overlap)
$\boldsymbol{TPP(a,b)}$	a is a tangential proper part of b
$\boldsymbol{NTPP(a,b)}$	a is a non-tangential proper part of b
$\boldsymbol{a=b}$	Equality
$\boldsymbol{NTPP^{-1}(a,b)}$	Inverse non-tangential proper part (b is a non-tangential proper part of a)
$\boldsymbol{TPP^{-1}(a,b)}$	Inverse tangential proper part (b is a tangential proper part of a)
$\boldsymbol{EC(a,b)}$	Externally connected (the closures of sets a and b overlap, but not their interiors)
$\boldsymbol{DC(a,b)}$	Disconnected

Table 3 – description of spatial relations in RCC-8

Randell also provides a transitivity table which describes the results of combining configurations so that sets of configurations can be constructed to describe more complex planned or expected spatial configurations which can then be checked for satisfiability as constraint networks.

Independently of Randell, RCC-8 equivalent models were also proposed by Egenhofer [EGENHOFER'91] and Smith and Park [SMITH and PARK'92] . This independent convergence on a single model and the results of Renz's configuration grouping experiments, which showed both a marked preference for grouping spatial configurations according to the categories of the model, both support the argument that RCC-8 provides an intuitive and cognitively-

adequate classification of spatial configuration. Additionally, Renz asked participants to explain why they had grouped configurations in the way they did, and this revealed that the three most often applied considerations in human classification of spatial configurations were topology (i.e. connectedness and containment), orientation and distance, the most popular of these being topology. These approaches are particularly interesting because reasoning about awareness, as Rodden's model in particular indicates, is strongly predicated on the ability to reason about different topological configurations between objects.

Unfortunately, Renz work in formalising RCC-8, and later work by Liu et. al [LIU and DANESHMEND'04] on QSR models concerning motion of robotic arms, highlights the difficulties in implementing these sorts of models as computationally efficient systems. Generalised consistency checking of constraint networks for RCC-8, for instance, is NP-Hard, although Renz does offer some tractable subsets. There are also fundamental categorisation differences to be considered between the various flavours of RCC and the Awareness model. Whereas RCC was conceived to describe the pairwise arrangement regions and concerns itself with neither relationships between regions and points nor higher-arity configurations of spatial arrangement, we can see from the descriptions of awareness modes in Rodden's model that relations may explicitly involve parts of regions and have arity of up to six.

Nonetheless, we would argue that the idea of qualitative spatial reasoning is useful in the domain of context-aware computing both in that it provides a formal basis for describing interaction that is often difficult to capture quantitatively and Renz's findings with respect to the significance of topological, proximate and orientive relations offer the potential to provide a solid foundation of reasoning and usability across the fragmented Mixed-Reality spaces that users now inhabit.

Seams

A qualitative approach is particularly useful for reasoning over representations which may be dynamic and partially known, since in quantitative reasoning it

can be particularly difficult to deal with uncertain or inexact knowledge. This aspect of context-aware computing, we believe to be important and under-considered in context-reasoning platforms. A traditional desktop computer environment provides a high level of memory, storage, and processing power, a (usually) stable network connection, and a small set of well-defined methods of accessing data like files. In a mobile ubiquitous environment, on the other hand, resources are limited. Hardware capabilities are heterogeneous, ranging from powerful smartphones to low-power wearable and embedded devices, access to data is frequently through the cloud or locally networked devices and at the mercy of unstable wireless connections, and sensing and communication & interaction between users, as Crabtree points out, can be fragmented and unreliable.

A traditional illustration of this unreliability is cellphone signal given in [CHALMERS'04] . Although carrier coverage is vastly improved from that of the 90s, most of us have had the experience of walking around with phone at arm's length in an attempt to recapture a dropped signal and popular events like fireworks displays can still overload a network's capacity to the extent that many users are unable to place calls during the event. Chalmers argues that because the underlying behaviour of the network is completely obscured to users, it is difficult for them to ascertain what to do when a call cannot be made. Is there a fault with the phone? The other person's phone? The network infrastructure? Is there no coverage? Is coverage over capacity? Or have they simply run out of credit? Handset manufacturers attempt to ameliorate the problem consists of illustrating signal strength on the phone's display, leading to the aforementioned signal-bar dance which is often about as effective as it's precipitation-orientated cousin. Some users, Chalmers finds, prefer to use utilities that display details of the broadcast tower they're currently connected to, making visible the hidden handoffs from tower to tower that our phones otherwise perform constantly without our knowledge. This can help them determine why they're experiencing problems. But as [CHALMERS and MACCOLL'03] points out, this technological shortfall has its benefits. If signals can be unstable and a user cannot know why or how, or predict when, the problem occurs, then neither can the people phoning her. Some users exploit

this uncertainty to avoid, or end, inconvenient calls by misrepresenting their signal quality. In fact, so popular are this tactic and its companion: entering an imaginary tunnel, that they have pervaded popular culture throughout the past two decades appearing frequently in films, TV shows and novels. These sorts of ubiquitous systems issues which straddle the intersection of uncertainty, instability, and resource access problems are known as seams.

Seams were initially defined by Weiser as something to be eliminated from Pervasive systems. His vision for ubiquitous computing was as a means to provide tools for the user to accomplish her tasks that are as transparent and intuitive in their usage as a hammer is to a carpenter, so that a user may "focus on the task not the tool"[WEISER'94] , This in contrast to desktop environments, which are criticised by McCullough as "too seldom [tapping] latent dispositions (skills we already have) and too often [requiring] arbitrary instruction." [MCCULLOUGH'04]

In searching for a stronger phone signal, not only is the user's focus directed away from her task, so too are her actions. Even where this sort of seam is exploited, the action taken by the application is still to bring the underlying technology to the fore. Weiser gives an example of seams arising where two heterogeneous technologies are forced to work together by reducing the functionality of each one until they find a common interactional ground. Supporting the seamless interweaving of widely different technologies whilst letting those technologies retain their defining characteristics is a difficult goal to achieve, he argues. Macoll [MACCOLL, CHALMERS et al.'02] , Chalmers and others have proposed that the best way to deal with seams is not to attempt to make them invisible, but to selectively reveal them to users, as signal strength and GSM utilities selectively reveal parts of the GSM infrastructure with the aim of increasing visibility of the behaviour and influences of the mobile phone signal. In fact, [CHALMERS and MACCOLL'03] demonstrates that users are likely to see and attempt to exploit seams, regardless of whether developers choose to deliberately present and leverage them. In Chapter 7 we introduce philosopher Gilbert Simondon’s theory of transduction in technological

systems and use this to discuss and analyses the processes, structures and behaviours involved in these approaches.

Exploitation of Seams

In their Mixed-Reality performance game Can You See Me Now? (CYSMN) online players moved their avatars around a map of the city, attempting to evade capture by runners who were tracked by GPS through the city's streets. Initially, runners were frustrated by the degradation of GPS signal in some areas due to the presence of tall buildings - urban canyons being a well-known GPS artefact - meaning the precision of their location updates dropped significantly. This lead to situations where a runner would believe she had captured an online player by running through their current location, but the runner's GPS would read them at approximately 10m before and 10m after the online player's location.

It should be briefly noted that this issue arises not only from the urban canyon effect but also from the implicit treatment of GPS - a digital and therefore fundamentally discrete sensing technology - as a continuous representation of a user's state. From the system's point of view, since the runner was only reported as being 10m before and then 10m after the player, they must have jumped from the first point to the second without passing through those lying between.

A fundamental goal of MacColl and Chalmers' work was to explore how users react to this sort of mobile computing issue and they discovered that as their knowledge of the game's Mixed-Reality geography (that which combines the physical world, the online players' virtual world and the precision-variable world of the runners' GPS) grew, they began to appropriate these seams in the game's ecology to give themselves a tactical advantage. Whilst online players were shielded from the GPS problems, this naivety led them to move anywhere in the gamespace without preference. The runners, on the other hand, began to refuse to chase players into areas of low positional precision, instead lying in wait to ambush them when they emerged into a more responsive game region.

Managing Seams and Uncertainty

MacColl and Chalmers enumerate Benford's four strategies for dealing with uncertainty in Mixed-Reality ubiquitous systems. These are:

pessimistic - hide uncertain information
optimistic - act as if everything is correct
cautious - present uncertainty explicitly
opportunistic - exploit uncertainty

These are adapted in later work [BENFORD, CRABTREE et al.'06] to a set of strategies for approaching seams in general:

Attempt to hide seams completely - Many applications make use of error correction - such as Tarzia's markov localisation technique [TARZIA'11] which is used to hide drops in accuracy in Wi-Fi location sensing
Ignore seams - Many early context-aware applications in the literature do not detail any engagement with seams at all.
Present seams to the user where they arise - This includes applications and utilities such as the previously discussed GSM connection information utilities
Design applications in such a way that they or their users can exploit the existence of seams -This we see in games such as Tycoon [BROLL, BENFORD et al.'06] which is specifically designed to gamify the problem of low bandwidth in mobile applications.

Reeves suggests[REEVES, PRIDMORE et al.'06] that developer options for approaching seams, rather than separating into four distinct paths, fall along a continuum from complete management by the system, which shields the user from the seams of the technology, to complete management by the user where users are afforded the opportunity to directly control how seams affect their application experience. Two examples he gives of system-management are Kidsroom [BOBICK, INTILLE et al.'96] and his own Journey into Space [REEVES, PRIDMORE et al.'06] . In Kidsroom, CG characters guide children through a story, including guiding them to specific zones around the room to

enable them to act out parts of the narrative. Journey into Space is the interactive story of a spaceship flight to the moon, where children are guided through the narrative in the interactive story room by a professional storyteller who has some knowledge of how the room is designed and can incorporate instructions that avoid or mitigate some of its seams. On the other end of the continuum, where seams are completely managed by users, are the Seamful Game and Can You See Me Now. Seamful Game requires users to travel around their city collecting virtual coins which they then upload by connecting to Wi-Fi. Since a player connected to Wi-Fi can have his coins stolen by other players connected to the same SSID, players must explore and learn the geography of Wi-Fi signals in their city in order to improve their game. This aspect of the application creates a gamification of the seam that Reeves terms "the spatial character of the network".

Although seamful design is a very interesting area of Pervasive Computing with particular potential for exploitation of augmented and Mixed-Reality spaces in gaming, there is relatively little research concerning platforms that support the development of applications along the full extent of Reeves continuum and across the entire MacColl/Chalmers/Benford taxonomy of seams. One contribution of this work is to initiate the discussion of how a model and platform for reasoning about context might best achieve this, and offer a way of thinking about seams that allows developers to explore and respond to them more easily. Since Reeve's work highlights the spatial nature of seams and Benford's the importance of the aspect uncertainty, we hope that our spatial model which combines both quantitative and qualitative reasoning will be uniquely positioned to provide this support.

Conclusions

This chapter introduced the concept of context and through a variety of examples explored how advances in sensor technology and the increased popularity of social applications is creating a vast pool of heterogeneous, dynamic and partially known context elements and representations that

developers of context-aware applications must work with and reason about. With this in mind, we examined several approaches to representing and reasoning about context and how these deal with this heterogeneity and dynamism. As the solution proposed in this work is of a spatial nature, we then looked at several approaches, both quantitative and qualitative to reasoning about space. This included the Awareness system of Benford and Fahlen and its extensions by Rodden and Dix, upon which we base our own context-reasoning model. Finally, we gave an overview of the Ubiquitous design concepts of Seams and Seamfulness and their significance for Context-Aware applications.

In the next chapter, we present the concepts that form the theoretical basis of our model and discuss how they relate to existing seamful and context-aware applications.

Glossary of New Terms

- **Element of context/context element** A property of a user's state which forms part of their context in an application but may be represented in many different ways depending on how it is sensed or gathered. For example, location is an element of user context.
- **Context element representation** One of possibly multiple ways of sensing or gathering the data that specifies an element of the user's context. For example, GPS is a representation of user location as a point in a co-ordinate system annotated with various extra values such as altitude and number of satellites visible.
- **Model** Abstraction of a context-element's representation to a form used for reasoning within a context-aware application. For example, a model may abstract distances between GPS points to their Euclidean distance and discard altitude data and number of satellites visible.
- **Dynamic/Mutable** Those representations whose properties may undergo changes during the runtime of an application, altering the set of possible values they can take or relationships between those values.
- **Static** Those whose properties do not undergo changes during an application's runtime.
- **Partially known** Context representations whose structure and extent must be explored and discovered by developers. Frequently because they are derived from infrastructures owned by third parties and not tasked to provide the form of context they are being used for e.g. GSM cell tower IDs whose primary function is not to provide users with a location.
- **Fully known** Context-representations whose possible values and their relationship are entirely known to developers before runtime.

- **Qualitative Spatial Reasoning** Reasoning about how entities are arranged in space in a topological way without referring to numerical coordinates, locations or distances.

Chapter 3

The Theoretical SPACES Model

Introduction

The previous chapter discusses a selection of context-aware applications, and frameworks for approaching context. These applications illustrate how context can have many components, each of which can be drawn from the physical sphere, the augmented sphere – where real world objects are given virtual properties, accessories and abilities, the virtual sphere, or the social sphere. They also show that a single component of context, for example a location, can be sensed or created by an application in different ways. Thus, location can be detected as GPS, as Wi-Fi proximity, or as a user-defined label. Finally, they demonstrate that context is not always what is sensed, but can be derived from manipulating and combining sensed and gathered user contexts to create something new, for example in Can You See Me Now (CYSMN), the sensed location of players in the physical world is overlaid with the location of players in a virtual world to allow both players to experience each other as if they share the same space, and their context becomes their proximity to each other within that shared space.

Figure 4 – Context properties may be static or mutable with varying rates of change

We have also discussed how components of context can be static or dynamic/mutable. Static contexts being those whose set of possible values and the relationship of those values to each other are fixed and do not change. Mutable contexts being those where this set of possible values, or the relationships between those values is not fixed, because it changes during runtime. We also saw that mutable contexts can be highly dynamic, for

example social connections on Facebook to which millions of changes are made every day, or they can be of moderate to low dynamism, for example GSM cell tower networks in mature markets, in which tower IDs may change, but do so infrequently (Figure 4).

Ideally, developers of context-aware applications should be able to compare, combine and translate user states between different forms of context, and reason about any user's context both within and across these forms, with ease. If a user with GPS technology declares their location and another user declares their location through Wi-Fi, an application should be able to reason about the relationship between the two users, regardless of the difference in the underlying sensing technologies. If an application requires context components drawn from different spheres, it should be easy to bring together and combine those components. For example, a backpacker might want to find Couchsurfing [COUCHSURFING, '15] vacancies nearby, and prioritise those vacancies by the hosts who are closest to him both geographically and socially. Such a scenario is just one example of combining static context elements with mutable ones.

Applications should be able to make use of any mutable element of context and reason about it in relation both to other mutable elements of context and to static elements in the same way as they would reason about two static elements of context in relation to each other. And finally, a user's context should not be tied to a specific application or vendor, nor should developers constantly have to reinvent the wheel when gathering context. If one application has determined that its user is watching a film with several friends, that context should be available to other applications in a form that they can manipulate and reason about, without them having to gather and process the information to construct it from scratch themselves.

Current context-aware platforms, however, lack the means to do all of this easily. Where context is created from heterogeneous sensing technologies, it can be difficult to produce mappings between the sets of context elements. Many mobile apps still require power-hungry GPS to use location-aware features, rather than making use of other available technologies. Where two components of context are drawn from different spheres, it can be difficult to

determine if there should be a relationship between them, and what that relationship should be. There is no organised methodology or infrastructure for reasoning about the relationships between different types of context. An application serving our couchsurfer, for example, must combine and compare measures of physical distance with measures of social distance, the structure of the latter set being subject to constant change. And, where a user's context has been constructed in a particular application, it can be difficult for other applications that might make use of that context to access and parse it, or take advantage of any data or services that can be accessed using that context.

We believe users should be presented with a world of interconnected, augmented environments, which transparently support their activities, and developers should be able to provide these environments with the use of an open platform or platforms through which to gather, manipulate and reason about context. However, due to the lack of such a platform, users are instead presented with a set of isolated environments which do not support them as transparently and fully as they should. And developers are restricted both in the range and variety of context elements they can make use of, or forced into the onerous and often complex task of manually declaring relationships between elements of context before they can be reasoned about.

A platform that provides developers with the capability to easily create applications that reason about a variety of representations and types of context requires the underpinnings of a model. An appropriate model would allow for a structured definition of different types of context and we believe that this can be obtained by extracting their defining features and abstracting their structure to forms which support a well-defined set of processes for reasoning about context. In this chapter we present the theoretical bulk of our model which comprises of these feature extractions, structural abstractions, and reasoning processes.

The Spatial Nature of Context

We have seen that many context-aware applications depend heavily, even solely, on the physical, spatial property of location. We have also seen that

location can be described by myriad systems, from the Cartesian structures of GPS to the individualised labels users give places they are familiar with. Despite their differences, these systems of representation all describe the same underlying set of physical location states. And once we extract the features that define each representation, we find that those features are shared not only by other representations of location, but by many representations of other elements of context including those which are not location-related, and in fact by many representations of context elements which do not refer to spatial, even physical, properties of user state at all. Since these elements of context share defining features with those representations of location, which is by its nature spatial, we hypothesise that it is possible to abstract many, if not all representations of elements of context, whether spatial or not, physical or virtual, to sets of features that allow them to be reasoned about as one would reason about a space. For example, a graph describing social connections has the same basic structure as a graph describing GSM cell-tower adjacencies, and can be reasoned about using the same methods of comparing path lengths for equality and subgraphs for overlap. Although graphs of context-properties like social connection are not spatial in nature, for example they do not have an embedding in physical space, this abstraction will allow us to use the same reasoning processes and structures for all user context reasoning, thus simplifying our model.

We believe that reducing the differences between various elements of context to the set of differences between a few different types of space provides a number of advantages. Firstly, that it vastly simplifies the process of reasoning about the relationships between users who draw their context from different representations. Secondly, that it facilitates the adoption of new types of context, since it reduces the complexity involved in describing their relationship to pre-existing types of context. Thirdly that it allows meaning to be inferred and reasoned about more easily from different combinations of context elements. And finally that it simplifies a variety of processes for reasoning about mutable context elements and those whose structural details may only be revealed throughout runtime.

In this chapter, we examine a subset of context-aware applications, those in which a user's context is leveraged to support their current activity using information, services or behaviours. We will show that the elements of context that these applications rely on, and the way in which they represent those elements, can be modelled as abstract spaces which conform to one of four sets of properties. These properties allow us to construct a set of simple processes for describing and reasoning about context in these spaces, and these processes combine with the properties of the spaces in which we are reasoning, to form the model that will underpin our context-gathering, -manipulating and -reasoning platform.

Space: The Foundation of the Model

The role of user-context in a context-aware system is to describe the features of the user's environment that are relevant to the system in deciding how to support their current task. This context may comprise a single feature, such as in the tourism and sightseeing application, GUIDE, where user location is the only property that determines the information about landmarks and attractions to be delivered to the user. Alternatively, context can be made up of more than one feature, such as in mobile automation application Tasker [TASKER (4.7), '15] , where a large variety of sensed information including location, time and device orientation are used to trigger specific mobile phone behaviours such as automatically diverting calls to voicemail whilst driving. We call each of the features of user state that an application employs to support their activity, an *element* of the user's context, or a *context-element*.

Context can be drawn from a variety of milieu, including the physical world where it is often gathered by sensing technologies. Dix's analysis of approaches to Mixed-Reality sensing systems [DIX, FRIDAY et al.'05] determines that although we most often consider sensing technology to translate the real world directly into a data model, what those technologies actually do is translate real world properties into representational data systems, which are then implicitly mapped into a model by the application developer.

This distinction illuminates a set of issues often encountered by developers of systems that use sensing technologies, which stem from disparity between the user experience of the environment– the picture painted by the sensor data– and the user's experience assumed by the developer based on that sensor data. For example, the Hitchers project, a GSM-based location-aware game developed to explore the properties of GSM location systems, revealed many inconsistencies in sensed GSM data, where the cell a user is located in can be affected by factors such as the speed she travels at and in which direction, and the local weather conditions. In one case the mobile device of a user following a particular route on foot might report a different sequence of GSM cells to that of a user following the same route by car because, moving slowly gives her device the time to detect and connect to cells with better signal quality without the frequency of handovers degrading her service [DROZD, BENFORD et al.'06] .

Rather than reasoning about user context at the apparent representation level, our system emphasises the model level, providing developers with a framework on which to build their representation models, and providing any model built on this framework with a system of reasoning about user context both within that representation's model and across pairs of those models built on the framework. Using the Physical<->Representation<->Model [DIX, FRIDAY et al.'05] approach, taking GPS as a *Representation* of Physical location, an example of a mapping from the Physical world to the space of our Model via the Representation would be encoding The Tower of London to the GPS reading *51.507995, -0.076188* to the point `{51.507995, -0.076188 }`in the Cartesian plane defined by $\{ \forall(x, y) . -180 \leq x \geq 180 \wedge -90 \leq y \geq 90 \}$. We can also apply this approach to non-physical context-elements, for example identity as a function of social connection. Here our representation takes the form of a person's Facebook connections and we can map *Joe Bloggs* to *Joe Bloggs' Facebook friends list*, and finally to our model as a node labelled **`Joe Bloggs`** in a graph where nodes represent identities of Facebook users and any edge between two nodes represents friendships between those users Figure 5551.

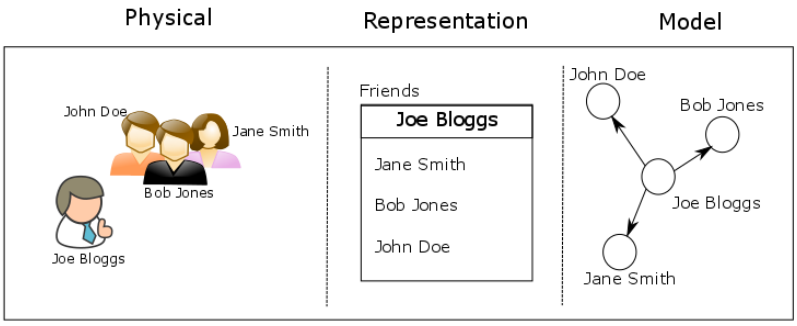

Figure 5 - Relationships between Physical, Representation and Model space when modelling social connection

A sensed context element's representation can be thought of as encoding physical experiences much as language does, that is, as a set of signifiers that can be decoded to specific experiences. For example, in a set of GSM cell IDs, each ID label signifies the physical reception area covered by the cell's broadcast tower. The abstraction of this set of signifiers to a set of points forms the basis of the space that our model uses to describe the representation. Each point of a representation's model can be decoded to a value in the representation, and in turn from that value in the representation to a specific physical experience. A model of the GSM representation might take cell IDs as the labels for nodes and create graph edges between them indicating adjacency between nodes. Thus a node in the graph would decode to a cell ID and in turn to the physical area that a user would probably be located in if their mobile device reported contact with that broadcast tower. Likewise, each physical experience encoded by the representation technology can be mapped to a point in the representation's model. Again this approach can be similarly applied to non-sensed elements of context, such as our Facebook example above.

Since many representations are so similar to their models, the question arises: why make the modelling process explicit at all? A mapping of location to GPS representation to GPS model, abstracts the set of all possible coordinates that could be generated by a GPS device to the set of points contained in the Cartesian plane bounded by {-180,+180}{-90,+90} Although this mapping appears trivial, it is important to note that differences exist. For example, the model uses a set of points that are continuous, whereas, due to the granularity

of GPS sampling, the representation's set of points is not. In not making explicit the changes and assumptions that their abstractions introduce, many platforms miss the opportunity to support developers in managing or even exploiting the seams created by their necessary abstraction of the representation.

Moreover, separating model and representation can help us see where representations that are fundamentally different can actually be modelled in the same way. A GSM cell tower network could be modelled as a weighted graph whose vertices represent individual cell towers, and whose edges represent the adjacency between two cell towers and the probability of a user transitioning from one cell tower to another. This structure is almost identical to the structure of the Facebook friends model described in the social connection example above. In fact, we as we progress through this chapter we shall demonstrate that modelling a variety of context-elements as spaces of some sort, be they graph-type, Cartesian, or other, is not only possible, but affords us several simplifications in constructing and reasoning about user context.

Modelling Context as Space

Points

Since we propose to model each context-element as a space, we can imagine that each possible value that the context-element can take is a point in that space. Sensing the value of a particular element for a user can be seen as giving the user a location in the context-element's space. If two users share the same context we assume that they are located at the same point in the space, and vice versa, that is two users with the same location in the space would be implied to have the same context in the underlying element. We can write this as

$$U_a(L) = U_b(L) \Leftrightarrow U_a(C) = U_b(C)$$

Where, $U_x(L)$ is the user's location in the space of the model and $U_x(C)$ is the value of their context.

We call the set of all known possible values of the context element's representation it's *point set.* In order to reason about these user locations within

the different point sets of our model, we must make certain generalisations about the properties of these spaces. To that end, we want to extract the features of those spaces that are most relevant to reasoning about context.

The underlying representations of graph-structured models such as our Facebook and GSM models raise particular issues when it comes to working with the concept of direction. With the exception of forwards and backwards, direction can often not be imposed on these models or must be imposed by adding additional structure as we saw in Chapter 2 with the combinatorial embeddings employed by Annotated Voronoi Graphs. However, in a model whose underlying representation supports a model with geometry, we can create the sense of direction – up, down, north, south, etc. – quite naturally. On the other hand, Graph-based models implicitly convey structure and containment whereas geometric systems need to be annotated with additional rules and information in order to do that. A model of a representation that lends itself to graph-structure could, for example, be a model of location using Wi-Fi network proximity. Its representation encodes a physical location as a hash of all the networks visible from a set of locations, and the model of that representation presents this as a graph in which an edge between two vertices indicates the physical adjacency of two locations. Since the physical area encoded by a particular node in the graph can be sizeable there is an in-built sense of containment in this model, but it is extremely difficult to determine a sense of direction, without augmenting this model with further structure such embedding it in a model of GPS. A context-sensing technology that could give rise to similar properties is an Active Badge-style representation of an office building, where users track their location by swiping smartcards as they move from room to room. In this model, rooms and corridors are encoded as vertices and doorways as edges between them. A model of the same office building which uses a GPS representation, however, would have to annotate the Cartesian planes of the building's floors with labelled shapes in order to achieve the same structure, or embed subsets of itself into the nodes of a graph-based room model.

Our approach to reasoning about user-context is to first give each user one or more points of location in these models, which represent their context. We can then describe their changing context through changes in their location within the model, and we can determine the application behaviour that their context should trigger by reasoning about their location in relation to the location of other users in the model. Since the different representations support different types of model and these different types of model place different restrictions and requirements on the processes used to reason about user context within them, we should separate the models into groups which support the same type of structures and therefore reasoning practices. One of the ways we do this is by separating the models of these representations into two classes: Continuous or Discrete.

A *continuous* model is a space whose point set structure is similar to some subset of $\mathbb{R}^*$, although it is not necessarily Cartesian in its geometry. A *discrete* model is one with a set of points whose properties are most similar to graph-based models.

Creating meaning with distance

The applications for which we designed our framework support user activities by delivering, or providing access to, particular Information, Services or Behaviours (ISB) in response to the user's context. We call an ISB bundle whose delivery or availability is triggered in response to a change in user context a *payload*, and we can think of some users as consuming these payloads, whilst others produce them.

When we examine a user's sensed or sampled context as it appears in our model, which is as a point in a model space, the point in isolation does not give us any indication of the payloads that should be delivered to the user. The GUIDE application, for example, is not simply concerned with a user's location, but rather what is in the vicinity of that user's location.

Just as location in the model does not have to map to physical location, but can represent a sensed or constructed value in a variety of physical and virtual context-elements, the nearness of one user to another in the model does not

have to refer to physical proximity, but may instead indicate similarity of circumstance. For example, social music service last.fm provides users with a list of musical neighbours, other users who have extremely similar music tastes to them to help them discover new music that fits their tastes.

Of course, nearness is an abstract and subjective concept. Our model, rooted in spatial properties, designates the concrete and objective value: distance, as the key feature of the relationship between two modelled user contexts. Given a distance between the context of one user and another in a particular model space, an application can then apply its own subjective criteria to decide if they are near to each other or not and which payload deliveries that nearness should trigger. For example in social location-sharing app Foursquare [FOURSQUARE, '16] , we can model the context-element location as space a graph of unconnected vertices whose names correspond to place names, and we can model the context-element of social ties as a graph whose vertices correspond to users and whose edges correspond to friending relationships between those users on Facebook. When a user Bob Schmitz checks into Tate Gallery, and Bob's Facebook friend Eric Smith checks into the same location, the pair can be appear to have similar circumstances – both visiting the same gallery. This should trigger each of them to receive a notification that their friend is nearby. When those contexts are mapped to the Social Ties space as points "Bob Schmitz" and "Eric Smith", and to the Place Names space as the point "Tate Gallery" it is clear that the two users are near to each other in both spaces [Figure 6662], and that nearness can trigger delivery of the same notification.

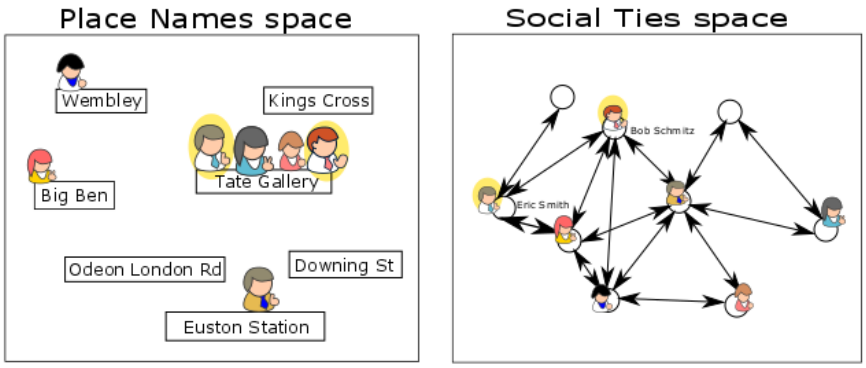

Figure 6 – Bob and Eric are the only two users close to each other in both the Place Names space and the Social Ties space

In order to enable reasoning about user proximity across different types of space, our model's definition of distance subscribes to certain rules. We adopted the concept of a distance metric from the field of metric spaces to describe distance over the set of points which forms the basis of a context-element's model. A distance metric applied to a set maps a pair of points to a real-valued number. Given a context-element's point-set S then, distance for the space constructed from S is described as a function:

$$d: S \times S \rightarrow \{\mathbb{R} \cup \infty\}$$

And for two points x and y in S:

$$d(x, y) = c \mid c \in \{\mathbb{R} \cup \infty\}$$

The rules we adhere to for *d* are:

1. $\forall s_1, s_2 \in S, \ \exists x \,.\, d(s_1, s_2) = x$

That is, d is defined for every pair of points in S. Recall that we mentioned partially known representations, i.e. those about which we may discover additional information at runtime, and mutable representations i.e. those which may add or remove points or distance information during runtime. We require this property to force developers dealing with these representations to provide a behaviour for all possible inputs to the distance metric equation whether a true distance has currently been determined for each case or not. We allow that this value be any positive Real number including infinity, which we suggest be

employed to indicate where one point cannot be reached from another, or where it is not yet known if one point cannot be reached from the other.

2. $\forall s_1, s_2 \epsilon S.\ d(s_1, s_2) \geq 0$

The function *d* always maps to a value of zero or greater. Just as negative distances do not exist in physical spaces, we require this because the notion of proximity in space, which we use to model contextual relationships breaks down with their existence.

3. $\forall s_1, s_2 \epsilon\ S, d(s_1, s_2) = 0 \Leftrightarrow s_1 = s_2$

If the distance between two points is zero, then the two points are in fact the same point and the distance from any point to itself is always zero. This allows us to identify reliably when two users share a location and as shall be explained, supports the reasoning about user context across representation models. Nb. This prevents the creation of pseudo graphs similar to those employed in Hierarchical Annotated Voronoi Graphs where an edge can represent an entire subgraph, but this is because our model treats edges as structurations upon a space of location points rather than as locations themselves.

4. $\forall s_1, s_2, s_3 \epsilon\ S.\ d(s_1, s_2) + d(s_2, s_3) \geq d(s_1, s_3)$

Also known as the triangle inequality, this states that given 3 points in a set which construct a triangle, the sum of any two sides of the triangle will always be greater than or equal to the length of the third side. The purpose of this rule will be covered with our discussion of the mathematics of the model in Chapter 4.

We also include an optional 5th rule:

5. $\forall\ s_1, s_2\ \epsilon\ S.\ d(s_1, s_2) = d(s_2, s_1)$

That is, the distance between any pair of points in S is symmetric.

If a metric obeys this rule, we call the space symmetric. If it does not obey the 5th rule, we call the space asymmetric. Spaces which are asymmetric have

additional structure inherent to them, in much the same way as spaces which are discrete do. One use of asymmetry is to enhance developer control of access to particular resources. For example, a secret room can be created in a directed graph representing an office block by removing all edges that lead into the vertex representing the room. No user located outside of the room will ever gain proximity to those located inside. We shall further discuss asymmetry and the challenges it poses in Chapter 4.

Criteria for Proximity

Adopting the properties of mathematical spaces in our context models allows us to standardise how we think about changes in user context, and in the variables which affect that context. A change in a user's context is most often reflected as a change in their location in the model space, as in music streaming and discovery system Spotify where creating a new similar artist playlist based on a particular artist can be modelled by SPACES as positioning the user at that artist's node in a graph where each vertex represents a musical artist and each weighted edge between vertices represents the existence of similarity between two artists and the strength of that similarity. The playlist can then be dynamically generated from music tracks located at the vertices in closest proximity to the user. We can therefore imagine users with changing contexts as moving through space, and speak of them moving towards and away from each other as their context changes whether or not that context is their physical location. We can also use the same descriptions for those contextual changes regardless the type of context of which we speak.

In some spaces on the other hand, it is not user movement that reflects change in context, but a change in the space itself, or a change in the application's criteria for proximity to that user. In application Foursquare where social connections are used as part of user context, our model would place each user at the node that maps to their Facebook account. Changes their context – beginning new friendships, or ending old ones, would be reflected by changing the edges connecting that node to others, adding new connections to represent new friendships, and removing connections where the relationship had ended. As the application also leverages the activity of people with similar tastes to

provide recommendations for restaurants etc. each user could also be positioned in a graph where edges connect people who have visited and enjoyed the same venues, with lower weighted edges indicating a higher number of shared favourite venues and higher weightings indicating that the two users share only a few favourites. In this case changes in the weightings might also occur.

The GUIDE application for sightseers uses physical location as user context and information about landmarks and tourist attractions as support for user activities. As we have discussed, specifically, GUIDE is concerned not with where a user is, but what else is near that location to determine the information that is delivered to each user. When a landmark is within a specific radius from the user, the user can view that landmark's information on her mobile device. The distance from the user within which landmark information is delivered can be thought of as the user's *area of interest*. In some applications users can control what we term their area of interest, for example, work from Nijholt et al. [NIJHOLT, ZWIERS et al.'09] building on the AMI project [MCCOWAN, GATICA-PEREZ et al.'05] uses video analysis to determine gaze direction in order to create a more realistic Mixed-Reality meeting experience. The user here controls their area of interest – the audio and video feed they are most interested in experiencing – simply by moving their head.

Coupious [COUPIOUS, '11] , also employs user location as context. Whenever a user walks within a certain radius of a participating store, coupons and special offers are delivered to their phone, encouraging them to visit the store. In this case, it is the retailers who control the radius from the stores within which shoppers will receive offers. This physical area within which the store can advertise through the app to passing shoppers can be thought of as the store's *area of influence*.

In fact, even when proximity is not explicitly used as a property of user context, we can often still model the events that trigger the delivery of a payload to a user as proximity events. For example, Take Me with You [FUJINAMI, YAMABE et al.'04] is a system that reminds business people to bring their umbrella when they leave the office for a meeting on days when

rain is predicted. The action of determining if a user has a meeting is performed by analysing his calendar. The calendar can be modelled in our framework as a single dimensional space, i.e. a line, in which each point represents a point in time. Meetings would then be placed in the space at the point which maps to their scheduled time and the user's position would move along the line as the current time advances [Figure 7773]. When the user is detected to be leaving the office in the physical location context-element and their position in the calendar space is sufficiently close to a meeting object, the system would receive instruction to examine the weather to determine if an umbrella is required.

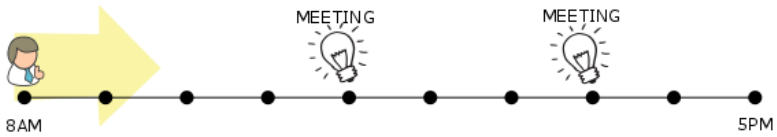

Figure 7 - The user moves through the single dimension Time Space towards meetings scheduled later in the day

The approach to modelling interaction between users as the interaction between their areas of interest and influence, called Awareness, was conceived by Benford et. al to support realistic interaction between users in virtual- and Mixed-Reality systems. All entities in a system that support interaction, whether they be users or virtual objects, have an *Aura* consisting of a *Focus* – their area of interest—and a *Nimbus* –their area of influence. The user's area of interest, their Focus, can be thought of as representing the area that they pay attention to and their area of influence, their Nimbus can be thought of as representing the area that they act on. Modelling context as the interaction between areas of interest and influence allows both what we call *Producers* – those who create and distribute ISB – and *Consumers* – those who receive ISB – to have input into the delivery of ISB payloads. As we have seen in the examples of Coupious above, this is important as it is not always the user who entirely determines her own context. This tension between a Producer who attempts to control distribution of a payload through her Nimbus size, and a

Consumer who attempts to control reception of payloads through her Focus size captures Dourish's view of context as a thing not sensed but continually negotiated between two or more parties in an application. We believe that as context-aware augmented reality applications proliferate and grow more complex, the ability of platforms to support negotiation between Consumers and Producers, rather than modelling just observation behaviour as occurs in GUIDE, or broadcast behaviour as occurs in Coupious, will become much more essential.

Presence

In order to reason about a user's context, we assign the application's chosen context-element to a model space which abstracts the representation used to encode that context-element. We give the user a location in this space, and this location is a member of the space's point set. We also give the user an area of influence and/or interest – Nimbus and Focus respectively and it is calculations on the interaction between one user's Focus and another user's Nimbus, performed using the model's Distance Metric, that we employ to reason about user context.

Since a point set in our model derives from a particular representation of a single element of context, and since context elements frequently have more than one representation – more than one method or technology for encoding their values—a user may be located in more than one space across our model. This is either because the user has access to more than one representation technology for that context element, e.g. using GPS and Wi-Fi for positioning; or because their context-aware application is drawing on multiple elements of context to support their activities. We can see an example of this in android app Tasker where users can combine time and location states to trigger behaviours on their device.

As discussed, in order to reason meaningfully about a user's context, their area of interest and/or influence must be defined along with their location. For a user with multiple locations, (i.e. a location in each of two or more spaces and/or multiple locations in a single space, for example when a user wishes to

produce content in one location but it interested in consuming content from other users at another location) this means a Focus and/or Nimbus must be defined for each of those locations. Users do not necessarily need to have both Focus and Nimbus defined at each location because a user may only wish to consume payloads, producing nothing which affects nearby users, in which case a Nimbus is not required; or they may only provide payloads, consuming none of those provided by the users around them, and in this case a Focus is not required. We call the user's behaviour at a particular location their *intent* and this can be either *Production*, *Consumption*, or *Both*. A producer's intent also encapsulates the ISB that they provide at a specific location. This allows different services to be provided at the same location by the same Producer, but under different areas of influence (Nimbus).

This distinction between Producer and Consumer is maintained in order to support modelling some real-world situations such as a smart kettle which tweets a notification every time it boils. This device *produces* information but is not designed to support *consuming* it. Tweeting at the kettle has no effect. Similarly, users may consume information from a real world event that they are physically located at and disseminate that information through what we consider to be a different space like a social network, for example taking a photo of a concert they're attending and sharing it on Instagram [INSTAGRAM, '14] . It is in this way that users and objects can possess multiple locations and these locations can be associated with different intents. For example, a user may have a GPS coordinate location, a street-address location, and also virtual locations within the multiple different social media networks that they use. At each of these locations the user may only consume information, only produce information, or do both.

Applications like Behavio and funf [BEHAVIO, '12] produce multiple streams of data about a user which can be accessed as context. These include location, environmental temperature, etc. In our model the value of each these streams would appear as a location for the user in a space within the model and each location would have associated with it either a Nimbus influencing the scope with which that information is allowed to propagate, or a Focus influencing the

distance in the model within which other entities can interact with the user. For example, a model of location might place the user at a node of a graph as a Consumer with a Focus that influences which of the entities in the surrounding nodes can deliver ISB to her. A model of the weather stream on the other hand might place the user in a Cartesian space, which models GPS coordinates, as a Producer with a Nimbus which influences how far information about the weather the user is observing can propagate.

From this, we see that using multiple locations, a user may be a Consumer in some parts of the model and a Producer in others. For example, we can define a Projector as one who collects information in one part of the model and rebroadcasts it in another and a Homeostatic Feedback Loop where information produced by one user is consumed by a second and used to manipulate the information which that second user produces, which is in turn consumed by the first user and used to manipulate the information which it produces

Modelling systems such as Behavio, there may be times when we want to reason about context as it derives from a single data stream, times when we want to reason about context as it aggregates across all available data streams, and times when we want to reason about context as it aggregates across a subset of these streams. This means reasoning about a single (Location,Aura,Intent) tuple; the set of all (Location, Aura, Intent) tuples and some subset of (Location, Aura, Intent) tuples for a particular user, respectively. In other applications as well, due to the flexibility of location/aura configurations, there may be times when we want to reason about several of a user's locations, aura and intents in aggregate, and times when we want to reason about them individually. To clarify this concept, we introduce presence. A user or entity's presence in the model encapsulates all of their locations, areas of interest, areas of influence and any information, services, or behaviours they provide access to or consume. Following the Dix model, presence in the "physical" world is the combination of the value of the entity's entire contextual state, coupled with their ability to provide or consume information, services and behaviours under certain contextual conditions and the circumstances under which those information, services and behaviours are

provided or consumed. In our Model it is a combination of a location(s) the information, services and behaviours they can provide or are interested in, and the aura that govern that provision.

Because a user can have multiple locations, and aura and intents with those locations, we call a (location, aura, intent) tuple a *Point of presence*. We are particularly concerned with supporting reasoning about context across different representations of the same context element. Both a user's ability to gain Points of presence in new model spaces and the model's ability to combine her awareness across pairs of these spaces are key to this feature. Later in this chapter, we therefore show how our model allows us to give a user a second point of presence in the model of representation which they would otherwise be unable to access, e.g. locating a user whose device only supports Wi-Fi in a GSM model space. And, given two users, both with points of presence in two representations, we detail a method for reasoning about the interactions between their aura in each of the two spaces and aggregating the semantic and quantitative results of that reasoning in order to give an overview of their contextual relationship as it exists across multiple context representations.

Awareness

Our model is particularly concerned with reasoning about context-aware applications in which user activities are supported or enriched through the delivery of payloads dependent on user context. We can say that in these applications context is used as a tool to manipulate the flow of payloads from those users or entities that produce them to those whose activities are supported or enriched by them, i.e. those who consume them. For example, the application GUIDE uses information about sites that are interesting or of historical significance to support the user activity of sightseeing. The user's proximity to a site is what triggers the flow of this information to her digital tourguide device. In this way the user's context in the app—her location—is used to trigger and manage the flow of information that she then consumes. In Tasker, a user may specify behaviours for her phone to enact under specific circumstances. For example, she might wish all calls to be diverted automatically to voicemail when she is located at work and has her phone

turned face down. In this case, the behaviour provided—diverting calls—which supports her activity—concentrating on work—is cued by her context, and that is her location and the orientation of her phone.

Social music service Last.fm suggests new artists for users based on those to which they have previously listened. Data about which bands each user listens to is aggregated across the entire userbase to produce similarity measures between every band. Given any two bands, the larger the proportion of the users who listen to both bands rather than just one or the other, the more similar last.fm decides that the two bands are. The more of a band's similar artists a user has listened to, the higher it appears in their recommendations list. This helps users to fill the important gaps in their musical experience by finding bands most similar to the ones they already like. Here, the proportion of similar bands to which a user has listened can be thought of as the user's proximity to a particular band. In this instance, it is not just which bands a user is in proximity to that determines the information the user receives – their personalised music recommendations – but the degree of their proximity to any given band.

Given a particular band, say Blur, the more bands similar to Blur that the user has listened to, the higher up their recommendations list Blur will appear. If we were modelling this, we would say that the closer the user is to Blur in the model, the higher up their recommendations list the band would appear, and this degree of closeness would be determined by the number of bands similar to Blur to which the user had listened. Another example, rooted in the physical world, appears in myriad augmented reality apps where virtual augmentations added to real world objects appear to grow larger as the person viewing them draws closer to the real world object. It is clear from these examples that it is not enough for platforms which reason about context to simply determine if one user is in proximity to another or not, we must also be able to determine the degree of that proximity, i.e. a quantitative measure.

Context-aware social media app Highlight provides users with a stream that notifies them of anyone nearby who they are socially connected to. A user may be socially connected to another because they have met previously, because

they share one or more Facebook friends, or because they have both declared an interest in the same topic. If two users are not socially connected they will not appear in each other's' feeds, even if they are right next to each other physically. Users also have the option to go invisible when out and about, meaning even if they are socially connected to someone nearby, they will not appear in their feed. If we wish to reason about user context as it operates in applications like Highlight, we must create a model which allows users who are invisible to see other users nearby, but not be seen themselves. One approach is to create an asymmetric model of the representation which makes invisible users infinitely distant from others in one direction whilst preserving their proximity relationship to other users in the other direction, so that they can see users they are physically near to but those users cannot see them. However, a simpler solution arises in the interaction of Focus and Nimbus.

Since it is the interaction of a Consumer's Focus with a Producer's Nimbus that allows payloads to flow from Producer to Consumer, we can concentrate our analysis of context on the configurations of Aura that occur when a producer's Nimbus interacts with a consumer's Focus. Classifying these configurations allows us to provide a qualitative measure of proximity in addition to the quantitative measure, and this qualitative measure describes the nature of the interaction between two Aura. This allows us to model an application like Highlight by hiding users from each other when they are in some Aura configurations, and revealing them to each other when they are in others. The size of both users' Auras combine to act as a negotiation on the sense of scale to be applied to their interaction.

In the MASSIVE system Benford et. al. call the result of this negotiation between Auras, *Awareness*. And it is employed to amplify or attenuate the transmission of video, audio and text feeds between users in distributed and co-located Virtual Reality (VR) spaces. In our system, where a user's presence is a model of their context, we use Awareness as both a quantitative and qualitative description of interaction between two users' points of presence. Awareness classifies the nature and degree of incidence between two users' contexts, or alternately between the context of a user and that of an object—virtual or

physical—in the system. In effect, it determines just how their contexts control the flow of payloads from producer to consumer. By manipulating her awareness, through changes in her presence, a consumer can change the payloads that she can access from each Producer, and by influencing other users' awareness, through changes in her presence, a producer can change the payloads that she can deliver to each consumer.

In Chapter 2 we described the qualitative topological reasoning model RCC-8 which can categorise different spatial configurations between pairs of convex 2D figures. Although it shows the utility of qualitative approaches to spatial reasoning and the importance of topological relationships, RCC-8 both contains some spatial configurations we feel are redundant for our purposes of modelling interaction between pairs of user presence. For example, differentiating between the configuration $A\ is\ Tangential\ Proper\ Part\ of\ B$ indicating that the figure B contains the figure A and that A touches the border of B, and the configuration $A\ is\ Non-tangential\ proper\ part\ of\ B$ does not seem to add any useful nuance to our model of presence and Awareness. Furthermore, since it is concerned not with presence but with simple convex figures, RCC-8 lacks the nuance to describe how Auras relate to particular points inside each other. For example, when reasoning about a Consumer's Awareness it is useful to know whether her Focus covers the Location of a Producer of which she is aware in addition to her Focus' intersection with the Producer's Nimbus.

Therefore, our SPACES model defines its own five qualitative types of Awareness which correspond to five possible configurations of Focus, Nimbus and Locations and roughly to five types of interaction between a Producer and Consumer. The first of these is *No Awareness*. In this case, the Consumer's Focus does not overlap or interact at all with the Producer's Nimbus. There can be no flow of payload from Producer to Consumer in this instance and the quantitative value of awareness in this configuration is always 0. For example, if we were modelling an application like Highlight, two users on opposite sides of the city, whose Foci and Nimbi only extend to the buildings they are in, would have no Focus/Nimbus interaction and would therefore have No

Awareness. Consequentially neither user would receive updates, from the Highlight app, mentioning the other. In our model we let AQl refer to the Consumer's *Qualitative Awareness* of a Producer—the type of Awareness she has—and AQt refer to her *Quantitative Awareness*—the value of that Awareness. A No Awareness is defined as

$$AQl = None$$

$$AQt = 0$$

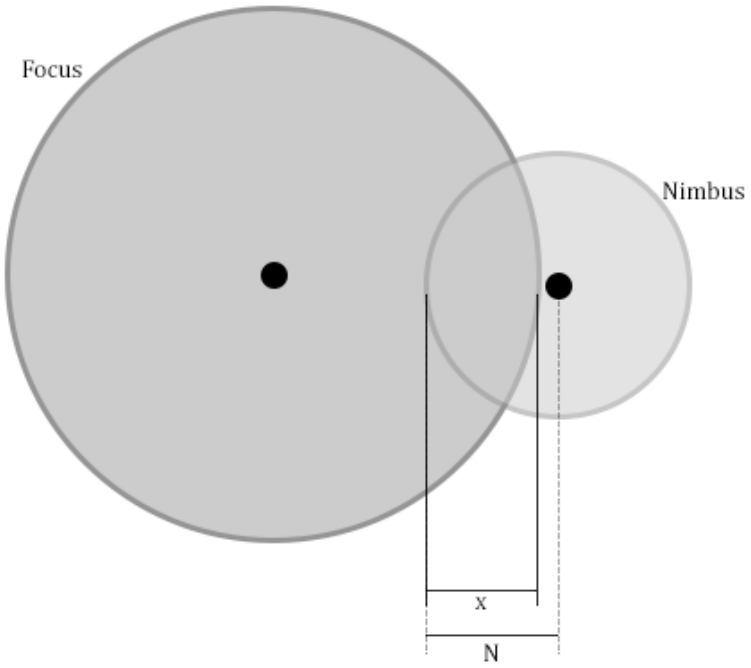

Figure 8 – In Low Awareness calculation, the projection of the Focus into the Nimbus is the size of the max scalar overlap x as a proportion of the Nimbus size N

The second type of awareness is *Low Awareness*. In this configuration the Consumer's Focus overlaps with the Producer's Nimbus, but the Producer's Nimbus does not interact with the Consumer's location neither does the Consumer's Focus interact with the Producer's location. The quantitative value of awareness in this case is derived from the depth of the projection of the Focus into the Nimbus and vice versa. In this case there may be a flow of payloads from Producer to Consumer which is manipulated according to the type and value of the awareness. For example, location aware gaming app Parallel Kingdom [PARALLEL KINGDOM, '13] , lets users build virtual kingdoms

that are overlaid on their physical environment. Since it is a massive multiplayer game, one user may have a location context that enables them to see the virtual structures another user has created in the game, however their context may not place them close enough to interact with that user. This can be modelled as Low Awareness, where some of the effects of an entity's influence on the environment can be observed by a user because they fall within the user's area of interest in the environment. Low awareness is defined as

$$AQl = Low$$
$$AQt = [Proj_F(N), Proj_N(F)]$$

Where $Proj_F(N)$ is the projection of the Focus into the Nimbus, that is the maximum scalar distance that the focus extends into the Nimbus as a proportion of the specified size of the Nimbus (Figure 8), and $Proj_N(F)$ is the projection of the Nimbus into the Focus, which is similarly the maximum distance that the Nimbus extends into the Focus as a proportion of the Focus' size.

The third type of awareness is *Passive awareness*. In this configuration the consumer is within the Producer's nimbus, however, the Consumer's Focus does not overlap the Producer's location. The quantitative value of awareness in this case is derived from the combination of its maximum -for the projection of the Nimbus into the Focus- and the depth of the projection of the Focus into the Nimbus. Although a Nimbus that covers the user's location does not necessarily cover all of her Focus, we envision the user's location as the most important part of her presence to be interacted with and this can be thought of as the strongest point of her Aura so that when a Nimbus covers that Producer-controlled portion of her awareness is maximised.

In Passive Awareness, a Consumer finds herself within the area of influence of a Producer. This could be interpreted as a model of the Producer forcing the effects of their influence upon the Consumer, as happens in some location-aware advertising systems, with users receiving advertisements whenever their context is that they are within a certain distance of a venue, regardless of whether or not they are interested in receiving them. We would model this as

flow of advertisements commencing whenever the user entered the venue's Nimbus – indicating that the user is entering Passive awareness of the venue – regardless of the size of the user's Focus. Passive Awareness is defined as:

$$AQl = Passive$$
$$AQt = [Proj_F(N), Max]$$

For some system-wide constant Max

The fourth type of awareness is *Active awareness*. In this configuration, the Consumer's Focus overlaps with both the Producer's Nimbus and location, but the Producer's Nimbus only overlaps with the Consumer's Focus, not her location. The quantitative value of awareness in this case is derived from the combination of its maximum – for the projection of the Focus into the Nimbus – and the depth of the projection of the Nimbus into the Focus.

Active awareness can be conceived as the complement of Passive awareness. Here, we can imagine that the Consumer is interested in the area which contains the Producer. Nokia's context-aware augmented reality system provides users with virtual information overlays on the real world as viewed through their mobile device's camera. The physical area visible to the user through their camera can be modelled as their Focus, and once a virtual object enters this area, it is available to them in some way, from simply being visible to emitting audio, to allowing interaction, regardless of the size of its Nimbus. Active awareness is defined as:

$$AQl = Active$$
$$AQt = [Max, Proj_N(F)]$$

The final type of awareness is *High awareness*. In this configuration, the Consumer's Focus overlaps both the Producer's Nimbus and location, and the Producer's Nimbus overlaps both the Consumer's Focus and location. The quantitative value of awareness in this case is the maximum possible. This maximum value acts as a limit for the value derived from the depth that the Focus penetrates the Nimbus and vice versa. High Awareness is defined as:

$$AQl = High$$

$$AQt = [Max, Max]$$

Returning to the previous example of Nokia's Augmented Reality Cityview, imagine a virtual album poster outside a music store across the street, which is visible through the user's camera on their mobile device. The poster may allow users to listen to the album it is promoting, but only if they stand near to the store's entrance, which makes them more likely to enter and purchase something. When the poster is seen by the user from across the street, the situation can be modelled as the user's Active awareness of the poster. However, when the user crosses the street they enter the area within which it is possible to stream the album. Using their phone to view the poster at this greater proximity, creates the condition of High awareness which allows them to access the service of streaming the album.

In the case of a low awareness configuration where a Consumer's Focus very nearly overlaps the Producer's location and the Producer's Nimbus very nearly overlaps the location of the Consumer, the quantitative values of low awareness can approach the maximum awareness value. In the same way, an Active awareness configuration where the Producer's Nimbus very nearly overlaps the location of the Consumer, and a Passive awareness configuration where the Consumer's Focus very nearly overlaps the location of the Producer, will have a quantitative value approaching the maximum. Since High awareness and No awareness are the only two states with unique quantitative values, in the cases of Low, Passive, and Active awareness, whose range of quantitative values overlap, the type of awareness supplied by the model can be used alongside the quantitative value to better inform manipulation of the payload.

With Awareness, as opposed to simple proximity, the relationship between user context and delivery of payload becomes a negotiation driven by presence. This negotiation governs both delivery of the payload and its manipulation by both the consumer and producer. For example, in the advertising system which pushes the same advert and coupon to any user who walks within 100 metres of a venue, what is being negotiated is control of the payload. The augmented reality system in which virtual objects grow larger as the user approaches them

is using awareness to negotiate strength of the payload. A model of Highlight, which allows users to choose to hide themselves from view is negotiating visibility. And the social third-party couchsurfing app which allows a user to see hosts closest to them both physically and socially, negotiates order. This negotiation may commonly result in manipulation of the payload by the Producer, but it can also result in manipulation of the payload by the Consumer, or both.

Awareness across Representations

The system we have thus far described gives us a method of modelling user context as presence in a space, where that space is governed and defined by a set of rules which we call its symmetry and continuity. It also allows us to control when and how context-aware applications support user activities by triggering delivery of Information, Service or Behaviour Payloads from a producer to a consumer under certain presence configurations. We model these configurations as Awareness, which is derived from reasoning about the relationship between the presence of a consumer and a producer, and it is the value and type of this Awareness which triggers and manipulates the delivery of payloads from producer to consumer. The techniques presented allow us to reason about presence and Awareness within a single representation of a single context element; however, we are also interested in supporting reasoning under more complex situations.

Many context-aware applications use location as user context, and location is just one of the many context elements that has multiple representations. Using Dix's system of structuring the flow of information about a user's state into a context-aware application: $Physical \rightarrow Representation \rightarrow Model$, we can say that for the context element of location there exist multiple technologies which could sample a user's context and each of these encodes the user's Physical location into a different Representation, for example, GPS, GSM, and user-defined location labels such as those employed to collect local landmark identities in Google's Ingress, each encodes the same set of physical locations to different Representations and with different levels of meaning embedded. For example, a location label offers more meaning to users familiar with the

area than the same location described as a GPS coordinate, but a GPS system may provide more complete information about the distance between two places than a location labels system is capable of.

In order to reason about the contextual relationship between two users who have different location sampling technologies, which encode their Physical location to different Representative location values, we must have a way to connect the two Representations. We do this by creating a Model which allows them to inhabit the same aggregate space, as they do in the Physical world. For location specifically, there exist a few platforms which resolve some of these representations to the same model.

Drawbacks in Existing Approaches to Modelling Context

The Placelab geolocation project, discussed previously in Heterogeneity and Dynamism of Context, Chapter 2, provided applications to allow users to war-drive for data which describes how GSM networks, Wi-Fi access points and GPS coordinates overlap. Sampled data was uploaded to a database which could be queried by applications, for example to resolve a GSM cell tower ID to a set of GPS coordinates or vice versa. The drawback with this approach was the large investment of time and effort required by developers and enthusiasts to map these overlaps and maintain the mappings, particularly those involving Wi-Fi which can be highly volatile as new SSIDs are created and destroyed by smartphone users employing mobile devices as temporary hotspots. There was also a lack of support for estimating a mapping between two points which had not yet been sampled. Such a feature would have allowed, for example, a user with GPS to interact with one using GSM locations even if the GSM location of the second user had yet to be mapped to a set of GPS coordinates in the Placelab system. The absence of this meant that application developers must either construct their own reasoning systems to provide these functions or must accept that there would be times users would be unable to interact due to lack of Placelab data.

A more recent approach is Google's Location Server (GLS). Due to the proliferation of Android devices, the company can now crowd-source location

data through background services on those devices. These monitor and report the Wi-Fi and GSM IDs users encounter as they go about their daily lives, as well as GPS information where available. From this, the company is able to produce reasonably accurate location readings for applications unable to use GPS, for example due to power consumption constraints. This system may support translation between three popular location technologies, but application developers who wish to take advantage of emerging technologies such as Bluetooth and NFC locationing are left without such support. As the number of location sensing infrastructures being commercially invested in diversifies, e.g. Broadcomm's gyroscope and pedometer enabled location chip, and Microsoft's epitomic location recognition systems, it is vital that platforms provide methods which cost little in time and resources for integrating new representations of location with existing ones.

Another drawback of relying on a system such as GLS is that it supports only one aggregative model of location. Some applications, for example, Mixed-Reality conferencing platform AMI stitches together two non-contiguous areas of location to allow people in one area to access resources in another. Other applications may wish to designate two physically discrete areas as the same conceptually aggregate area in order to minimise spatial processing burdens, e.g. in augmented reality apps where high levels of object placement accuracy are not required, virtual objects can be positioned in one space with key geometric calculations for that space cached, and other locations mapped onto that space by way of quick transforms.

These sorts of applications are not supported by GLS and the stitching and reasoning about proximity in both the single and aggregate spaces must be done at the application level. In addition, as context-aware systems become more sophisticated, we see more applications that combine multiple elements of context, for example Tasker which combines device location with other elements such as time and device orientation. To support applications which use multiple elements of context we require a context-reasoning framework which allows developers to slot these elements together and supports reasoning upon the resultant aggregate model. Finally, our exploration of the properties

different Representations, has revealed that different representations of the same context element contain differing and often complimentary meanings in their structures, such as the previously discussed examples of asymmetric spaces vs symmetric spaces, and discrete spaces vs. continuous spaces. The potential to increase the encoded meaning of these structures, made available to both applications and users, by embedding one type of space within another makes a compelling case for pursuing the construction of a context-aware systems platform which supports such embedding.

Challenges in Aggregating Representation Spaces

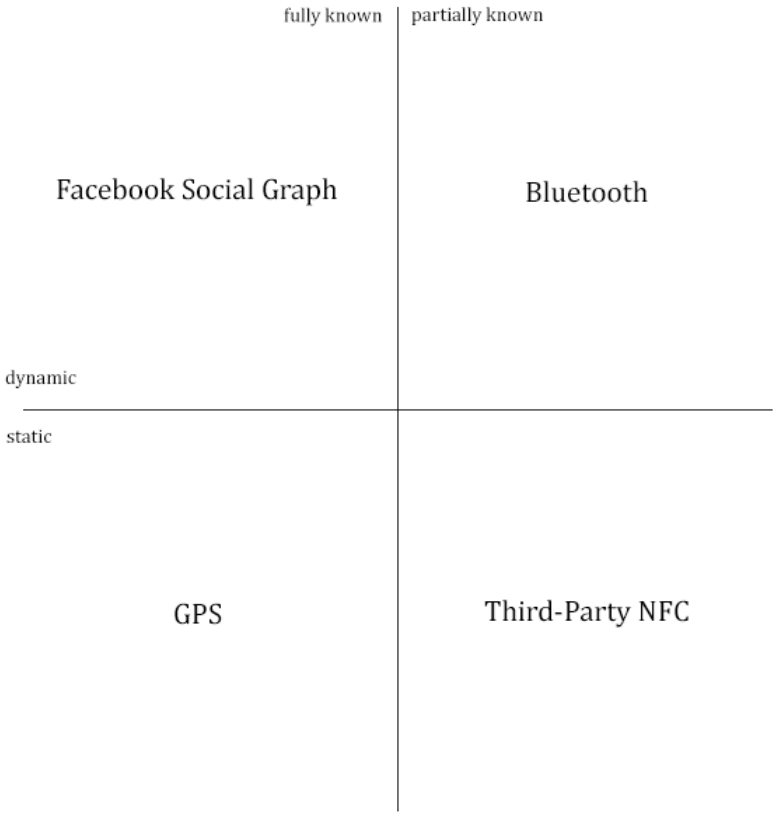

Figure 9 - Different types of context property spaces

When considering our approach to creating aggregate models, our two main concerns are the mutability of representations and their completeness. Context element representations can range from *fully known* to *partially known* and from *static* to *dynamic*. When a representation is fully known, all information about its current structure is understood by the Model. When it is partially known some of this information may be yet to be discovered and added to the Model. When a representation is static, the set of states underlying the

representation's model does not change although what the Model understands of them may change. When the representation's point set is dynamic, its underlying set of states is in flux and may grow or shrink over time, although whether the Model's information keeps pace with this depends on whether the representation is fully known or partially known. Furthermore, the known members of the static representation's point set do not change their relationship to one another over time, i.e. for all known members x,y d(x,y) is constant over time. When the representation is *dynamic* the distance relationships between existing members may change. GPS is an example of a fully known and static representation since we know the bounds of the values of any GPS location to be $\{(x, y)\}\ st. -180 \leq x \geq 180 \wedge -90 \leq y \geq 90$ and if the distances between each point are defined by a fixed distance metric, the Euclidean metric for example, their values for each position pair will never change. Bluetooth proximity is an example of a dynamic and partially known representation of location. The addition of new Bluetooth devices and the movement of other Bluetooth devices in and out of each other's range contributes to the alteration of both the distance metric and the pointset over time which makes the representation dynamic. The pointset may grow as more devices appear, or shrink as devices become inaccessible. In addition to this it is impossible for a developer or application to know every single Bluetooth device so the representation cannot be fully known and must be partially known. A static but partially known space is also possible. For example, a developer might decide to us a set of NFC tags installed in a physical space by a third party, by registering them in a database as they are encountered by users in a similar manner to how Hitchers registered its GSM cells. Though the tags may not change in location or number the third party may not have made the full extent and properties of the set public and thus is it only partially known to the developer. Figure 9 shows some examples of these property combinations.

Any framework that combines these representations must be able to support the full extent of this gamut of mutability and completeness. It is thus clear, that any approach to reasoning across two representations which depends on creating rules that align the structure of the two representation will fail when presented with representations whose structures are incomplete or subject to

change. Instead, we must look to methods which localise aggregation information as

much as possible.

Since the point is the smallest unit of context in our model, our approach is to employ a point-based aggregation system. When it is discovered that two points, each in a separate representation's model should be linked, we create between those two points a *Projection Point*. The existence of a Projection Point relationship between two points is an indicator that they both signify the same point in some physical or conceptual aggregate space.

A user's *native representation* is one in which they gain presence by the use of any sensing or gathering technology which they possess. For example, a user with a mobile phone can gain a native point of presence in the GSM representation, a user with a smartphone could gain native points of presence in both GSM and GPS, and a user with a tablet could gain native points of presence in both GPS and Wi-Fi. A native presence interacts with a Projection Point in its representation whenever its location is equal to a Projection Point or its aura contains the Projection Point. When a user's native presence interacts with a Projection Point, as its name implies, the Projection Point projects an *image* of the user's presence into the second representation. The location of this *projected presence* is dependent on the location of the Projection Point in the second representation, and the size of the projected aura is dependent on relationship between the Projection Point in the native representation and the native presence's aura. The closer the Projection Point is to the edge of the user's native aura, the smaller the projected presence's aura will be.

We have several rules for creating presence projections through Projection Points. Firstly, that a projection of the user's image cannot itself be projected through a Projection Point: only a native user presence can be projected into

another representation. This prevents situations where presence is projected back and forth between two representations creating an infinite mirror effect, and the more complex versions of this where infinite loops of projection are created across multiple representations. It also simplifies our model for the purposes of our initial testing. Secondly, a native presence can only be projected at most once into a specific representation and the Projection Point they are projected through must be the Projection Point that is closest to their native location. This is again to simplify our initial observations and to minimise any distortion between projection and native presence.

Since a user's projected presence is not a real point of presence, but an image of their native presence which has been manipulated and projected onto a foreign Representation, calculations of awareness between two users involving one or more projected presences are somewhat different to those involving only native user presences. First we calculate the type and value of awareness between each image-presence/native-presence pair, as if they were both native, then we aggregate those awareness-type results according to Table 433.

Some of these configurations may at first appear impossible. For example, the awareness pairing [Active, None] where a consumer's native presence has active awareness of producer's projected presence, but the consumer's projected presence has no awareness of the producer's native presence. In this situation, we might intuitively expect the consumer's projected presence to have passive awareness of the producer's native presence.

To understand why this is not the case, consider two users Eric and Mike, Eric consumes information using GPS whilst Mike relies on Wi-Fi for positioning data and produces information for Eric to consume. Let us assume that there exist two Projection Points P_1 and P_2 linking the Wi-Fi representation with that of the GPS. Projection Point P_1 links co-ordinates (x_1,y_1) with Wi-Fi node W_1. Projection Point P_2 links co-ordinates (x_2, y_2) with Wi-Fi node W_2. And further, let us assume Mike's Nimbus covers point W_1 in the Wi-Fi representation which allows him to project his presence through P_1 to the co-ordinates (x_1, y_1) in the GPS representation

Native Domain Awareness / Foreign domain awareness	None	Low	passive	active	High
None	None	Low	Passive	Low	Passive
Low	Low	Low	Passive	Low	Passive
Passive	Low	Low	Passive	Low	Passive
Active	Active	Active	High	Active	High
High	Active	Active	High	Active	High

Table 4 – Aggregating Qualitative Awareness Across Spaces

If Eric's native Focus in the GPS representation covers co-ordinates (x_1, y_1) Eric's native presence will have active awareness of Mike's projected presence, and we might assume that since Eric's Focus contains the location of Mike's projected presence and thus the co-ordinates of P_1, Eric's presence will be projected through P_1 to the point W_1 in the Wi-Fi representation, and because Mike's Nimbus covers W_1, as previously stated, Mike's native presence must necessarily have active awareness of Eric's projected presence, meaning the cross-representation awareness configuration [Active, None] can never occur.

However, this assumption is based on the false premise that there exists only one Projection Point, or that Projection Points create global symmetry between representations. Consider the second Projection Point in the GPS representation, the co-ordinate (x_2, y_2). If this is closer to Eric's native point of presence than point P_1 at (x_1,y_1), by the rules of projection above, Eric's presence will be projected into the Wi-Fi representation through P_2 onto W_2, not P_1, despite the fact that his aura does cover P_1. This will give Eric a projected presence located in the Wi-Fi representation at node W_2.

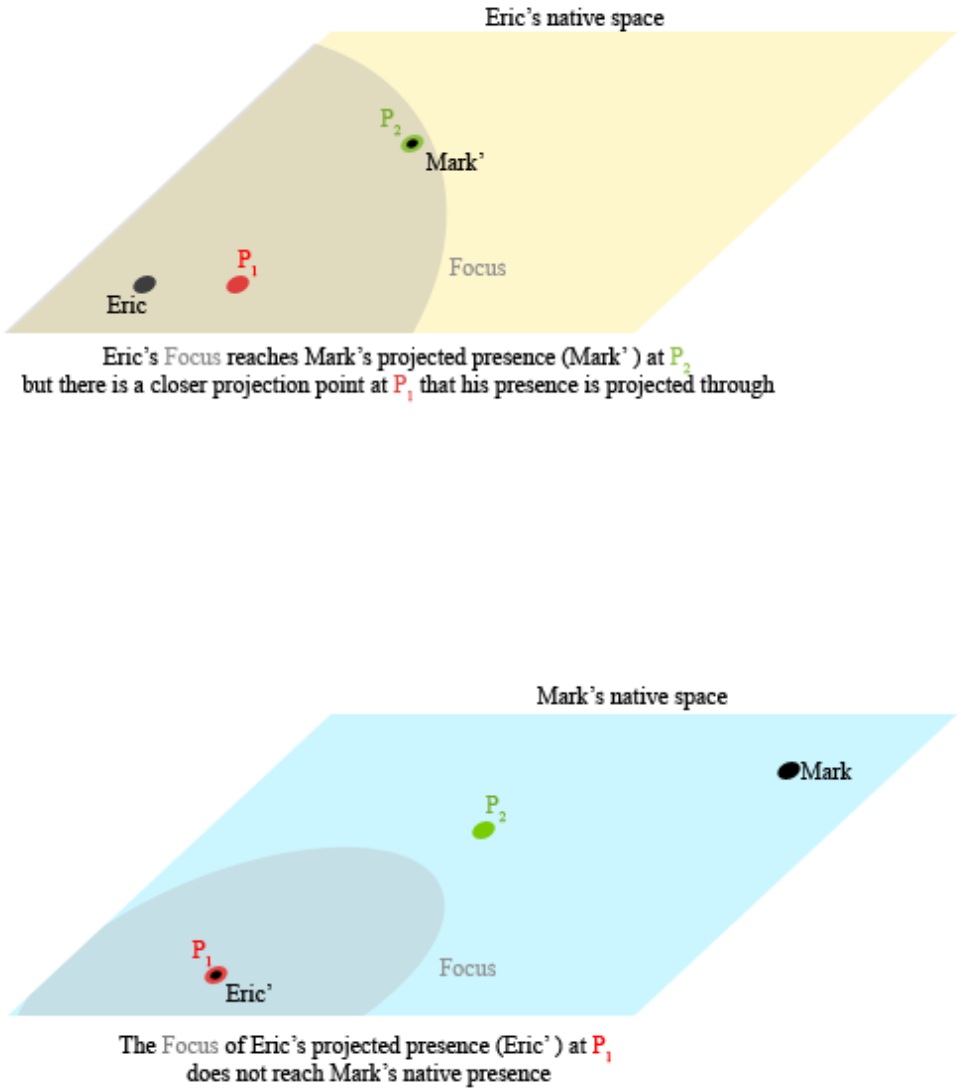

Figure 10 - Placement of Projection Points in different spaces influences Awareness Types across spaces

If Projection Points created global symmetry between two representations then the fact that Eric's Focus reaches far enough to cover co-ordinates (x_1, y_1), would imply that his projected Focus would reach far enough to cover node W_1, since (x_1, y_1) occupy the same aggregate space as P_1. This would allow his projected Focus to overlap with Mike's native Nimbus, since we already know that Mike's nimbus reaches far enough to cover W_1, and give Eric's projected presence at least low awareness of Mike's native presence.

However, the rules defining our distance metrics, allow that the distance between W_2 and W_1 could be greater than the distance between (x_2, y_2) and (x_1, y_1), even infinite. This is a reasonable scenario since we know that Wi-Fi

representations are frequently incomplete and thus there might be no data on the distance from W_2 to W_1, which the application developer might have chosen to represent with a distance of infinity, or guestimated to be extremely large. This means that even though Eric's Focus can reach (x_1,y_1) in the GPS representation, his projected Focus may not reach W_1 from the position of W_2 in the GSM representation, or indeed any node which is covered by Mike's native aura. This creates the situation where Eric's native presence has active awareness of Mike's projected presence, but Eric's projected presence has no awareness of Mike's native presence, as shown in Figure 10.

In this case the aggregated type of awareness is the same whether the awareness pair is [Active, None] or [Active, Passive], but when we introduce aggregation of awareness value in chapter 4, the ability to discern between these two awareness configurations becomes more important.

Conclusion

In this Chapter, we discussed how applications that employ user context to support user activity can have myriad definitions of the types of data which comprise context and the technologies used to gather and represent this data. We showed that despite all these differences, many different types and representations of context can be modelled as spaces and we presented rules for structuring and describing these spaces which categorised them into continuous or discrete and symmetric or asymmetric. We showed that once a spatial approach is adopted, user context can be encoded as presence in the modelled spaces and the states in which contextual activity support should be delivered can be modelled as awareness between producers and consumers. We discussed how different applications require different things from awareness making it necessary to be able to describe awareness both qualitatively and quantitatively. Activity support can then be provided as the delivery to consumers of producer payloads containing information, services and/or behaviours, and these payloads are delivered and manipulated based on the consumer's calculated quantitative and/or qualitative awareness of the producer.

We also discussed how the existence of multiple representations of the same context element, and the various demands of augmented and Mixed-Reality applications creates a need for systems that support reasoning about context across multiple representations. We explained how not all representations are complete and static at runtime and how this leads us to believe that a point-based approach to joining representations is most appropriate. We then described our approach which joins representations through pairs of points and supports reasoning across them using projections of user presence and aggregation of user awareness for native and non-native user presence points.

Thus far our model has been described in conceptual, rather than mathematical terms. In the next Chapter, we detail the mathematical processes involved in calculating and reasoning about presence and awareness both in a single representation and in multiple representations. We also investigate some of the issues that arise in reasoning about presence edge cases in particular types of representation, for example the conundrum of discerning Low awareness from No awareness in discrete asymmetric representations.

Glossary of New Terms

- **Point Set** The set of possible all values a user's state could take within a context-representation
- **Continuous** A context-element model whose Point Set is similar in structure to some subset of $\mathbb{R}^n$
- **Discrete** A context-element model whose Point Set has properties more similar to a graph-based model.
- **Payload** A bundle of information, services, or behaviours which users gain access to in given contexts.
- **Distance Metric** A mapping on a Point Set that gives the Real and positive distance between any two points in the set (which may also be infinity)
- **Symmetric** A Distance Metric which maps an unordered pair of points to a single distance value, indicating the distance between them is the same regardless of whether it is measured from the first to the second or vice versa.
- **Asymmetric** A Distance Metric which maps an ordered pair of points to a distance value and may map the same pair of points in reversed order to a different value, indicating that the distance between depends on the direction in which it is measured.
- **Area of Interest/Focus** The area a user pays active attention to
- **Area of Influence/Nimbus** The area a user acts directly upon

- **Aura** The user's Focus and/or Nimbus
- **Producer** A user with a Nimbus who wishes to distribute Payloads
- **Consumer** A user with a Focus who wishes to receive Payloads
- **Intent** Whether a user is a Producer, a Consumer or Both, and the Payload they distribute if any.
- **Presence** All of a user's locations, Auras, and Intents taken together.
- **Point of Presence** A single user location taken with its Aura and Intent
- **Awareness** a description and measure of the overlap between the Points of Presence of one user and another.
- **Qualitative Awareness** A description of the topological nature of the overlap between Points of Presence of one user and another classified as None, Low, Passive, Active and High
- **Quantitative Awareness** A numerical measure of the degree of overlap between the Points of Presence of one user and another.
- **Projection Point** A relationship between two points in different context-element models that indicates they signify the same point in some physical or conceptual aggregate space.
- **Native Representation** A context representation that a user gains presence in by use of sensing or context-gathering technologies.
- **Image/Projected Presence** A Point of Presence created for the user by the model in a non-native representation using its knowledge of the Projection Point relationships between the user's native representation and the non-native one.
- **Aggregated Awareness** When one user has awareness of another in both their native space and that of the other user (via their projected presence in the other user's native space) those awarenesses are combined to give a single overall awareness.

Chapter 4

The Mathematical SPACES Model

Introduction

In the previous chapter, we presented a conceptual model and methodology for approaching context reasoning as spatial reasoning. We described the properties and structures that can be extracted from different representations of various context elements, and how these properties can be used to define them as discrete or continuous and symmetric or asymmetric spaces. We showed how a user's context can be modelled as presence in these spaces and detailed how we use a system of Awareness to reason about the magnitude and type of relationship between any two user presences. Our model builds relationships between spaces by pairing points which it determines should represent the same point in some conceptually aggregate space, and we explained how these relationships allow users with presence in one space to project their presence into another space, effectively bridging the gaps in context-aware systems between users with different sensing or context-sampling technologies. We also described how our model aggregates a user's awareness types across two spaces. In this chapter, we explore our model further, presenting the mathematical calculations that comprise the presence and awareness reasoning systems of the model and discussing the issues that arise when calculating awareness within and across different types of space.

Calculating Qualitative and Quantitative Awareness between two native points of presence

In the previous chapter we presented the basis of our presence and Awareness reasoning system. Given any two users with the same context sampling or sensing technology, this system allows each user's context to be mapped into what we call a spatial model of the data representation that their sensing

technology creates, giving each user a *presence* in the space. A user's *presence* consists of a point, which often directly signifies the user's sampled context and an *Aura* – a Focus or Nimbus, which respectively can signify their area of interest or influence in the space, and thus the data representation. Users with a Focus intend to consume information, services or behaviours which context-aware applications use to support them in their activities. Those with a Nimbus intend to produce information, services or behaviours, called payloads, which support the activities of those users who consume them. We call these two types of user Consumers and Producers respectively.

The application Cityview, for example, allows users to consume information about events and offers happening in the area local to them. Venues, such as shops and cinemas produce information on film showings and special offers, and this is displayed to users as objects overlaying their view when they observe the venues around them through their mobile device's camera.

Modelled spaces

Given any Producer/Consumer pair with presence in a particular spatial model, we provided rules for determining the type of Awareness that the Consumer has of the Producer. That Awareness is used to notify the relevant Producer to deliver their payload to the Consumer. The Producer decides to trigger payload delivery based on the type and level of Awareness, and may also use these values to construct or manipulate the content of the payload such that it better supports the Consumer in her particular circumstance. In the previous chapter, we enumerated the 5 different types of Awareness, from None to High that can exist between a Consumer and Producer, as well as the conditions under which they manifest, for example, social location-aware application Highlight uses GPS locations to represent user context and can support social activities by delivering notifications to one user from others who are nearby. If Peggy's mobile device reports her location as 52.9500° N, 1.1333° W this is mapped into our model as the position (52.9500, 1.1333) in a Cartesian space. If Carrie's mobile device reports her location as 52.9600° N, 1.1233° W this is likewise mapped into our model as the position (52.9600, 1.1233). Then in order for Carrie to receive a notification from Peggy that she is nearby, the

pair's auras must form the configuration where Carrie's Focus overlaps Peggy's Nimbus or her location in the Cartesian space, and where Peggy's Nimbus overlaps Carrie's Focus or Location.

GPS sensing technology creates a representation that we model as a 2D Cartesian coordinate space with the distance metric d: $(C_L,P_L)\rightarrow \mathbb{R}\ x\mathbb{R}$ where C_L is the location of the consumer and P_L is the location of the producer in the space upon which the distance metric acts, and where the distance metric d is given by the haversine formula for distance between two points on a sphere:

$$d(C_L,P_L) = 2Rsin^{-1}\left(\sqrt{sin^2\left(\frac{x_c-x_p}{2}\right)+\cos x_p \cos x_c \sin^2\left(\frac{y_c-y_p}{2}\right)}\right)$$

R=6371 [the approximate radius of the Earth]

Applying the distance metric and point set rules given in the previous chapter we can determine that this is a Continuous and Symmetric space. Since we have given two types of spatial property: Continuity and Symmetry, every space we create in the model must have one of four structures:

1. Continuous Symmetric – a space whose point set S is Continuous and whose Distance Metric obeys the rule of symmetry: $d(x,y) = d(y,x) \forall x,y \epsilon S$
2. Continuous Asymmetric – a space whose point set is Continuous and whose Distance Metric does not obey the symmetry rule so that given two points P_1 and P_2, the distance from P_1 to P_2 may differ from the distance from P_2 to P_1.
3. Discrete Symmetric – a space whose point set is Discrete, i.e. can be mapped to some subset of $\mathbb{N}$, and whose Distance Metric obeys the symmetry rule. A discrete space may optionally include a second structure set E where if the point set $V \subseteq \mathbb{N}$, $E \subseteq V\times V$ or $E \subseteq V \times V \times \mathbb{R}$ and E indicates the set of edges between members of V which may be directed (members of E are ordered pairs) and weighted. Where a structure E exists, the distance metric obeys the distances between points defined by the structure of the point set.

4. Discrete Asymmetric – a space whose point set is Discrete, and may be similarly structured as a Discrete Symmetric space, and whose Distance metric does not obey the symmetry rule.

We have seen that the data representation of user locations generated by GPS sensors is easily modelled as a 2-dimensional Continuous Symmetric space whose limits are (+180, -180) in the x-axis and (+90, -90) in the y-axis. This model is adopted intrinsically by many modern location-aware applications. In contrast, the data representation created by the active badge system, an early location- and context-aware platform where users were tracked room to room in office buildings using badges with embedded RFID technology, can be modelled as a graph with undirected edges, where nodes represent rooms and edges represent doorways between those rooms. Since users can both enter and exit a room by the same doorway, this structure of discrete rooms and undirected connections between them is an example of a Discrete Symmetric space.

Wi-Fi location representations used by applications deployed on non-GPS enabled devices, such as laptops, may build their own maps of Wi-Fi environments by employing the same techniques as early adopters of GSM as a location technology did. Each unique set of visible SSIDs is considered to be one single location and a list of location adjacencies is built from the sampled user transitions between locations. For example, if a user's device reports seeing first the SSID set {a,b,c} and their next reported set is {c,d,e}, given a sufficiently small set time between reports, we can assume that the location signified by {a,b,c} is close to that signified by {c,d,e}. This representation can be modelled as a graph in a manner similar to the Active Badge representation, but since a change of visible SSIDs as given in the example only tells us that a user can move from {a,b,c} to {c,d,e}, we cannot guarantee that a similar adjacency exists in the opposite direction from {c,d,e} to {a,b,c} until a transition demonstrating this is sampled by a user. This requires us to introduce directed edges to the graph we construct from this representation, which makes the model of this data representation a Discrete and Asymmetric space.

Some location-aware applications, which make use of technologies such as GPS, require that additional information be encoded in the model and this can be done using the distance metric. For example, where transit time between points, rather than physical distance is the most important in the context of the services an application provides, a distance metric might map pairs of points to some value proportional to the estimated time taken to travel between them. Since transit time between two locations can differ depending on whether a user is travelling from the first location to the second, or vice versa, the resultant space in these situations can be both Continuous and Asymmetric.

As we shall see in the following sections the structural differences between these classes of space place restrictions on the possible structures of presence that we can employ, as well as influencing our methodologies for determining and aggregating awareness.

Continuity of context

We can imagine that context-aware applications support user activities by delivering situation specific information, services or behaviours to the user, alternatively we can say that they support user activities by limiting or filtering the information, services or behaviours that are available to the user based on her context, that in effect the context-aware application limits the user's view of the system in order to better serve their goals. In location-aware applications this limiting of the user's view occurs by filtering out payloads not local to the user, and in our spatial model of context this approach can be applied to all types of context element whether they be location-based or not.

Whilst limitation of view is useful, it can be desirable to provide cues to signify where further payloads beyond the user's field of view may be available and indicators of relative distance between users and objects. The experience of location-aware game developers in Savannah suggests to us that where hard boundaries of context exist without these cues, users can find the restriction of view confusing. User experience with Savannah, a co-operative location-aware game in which players take the role of predators hunting virtual animals in a physical gamespace, especially highlights this problem. Since users could only

"see" animals if they were stood within the animal's Nimbus, co-operating groups of players experienced instances where some members were stood inside, but at the edge of, the animal's Nimbus and thus able to see the animal, whilst others who were stood next to them at the edge of the animal's Nimbus, however in their case outside it, could not see the animal. Without cues to indicate to the players stood inside the nimbus that they were at the furthest point from the animal that would still allow them to view it, the players could not understand why some of them could see the animal and some could not, and did not know what to do to bring the animal into the other players' field of view.

Whilst the applications we are concerned with do not necessarily make context nor its relationship to the support the user receives for their activities explicit to their users, those users are nonetheless often aware both that they can influence the behaviour of a context-aware application by changing their own state and of how this can be done for specific goals. This can be quite naturalistic, mirroring physical world behaviours, for example, walking closer to a virtual billboard in an augmented reality app in order to view it in more detail, or it can be less so. However, we feel the user experiences in Savannah demonstrate, applications where small changes in a user's state produce comparable changes in the response they receive often provide the most straightforward experience for users. We call this property *continuity* of context and we believe it is important for a model that enables reasoning about context to support this continuity throughout its structure. For us, this entails ensuring that unless developers specifically intervene to the contrary, user awareness is locally continuous given changes in user presence.

Defining Aura

Since a user's awareness models her context, what we require of a user's presence is that it model the sampled data that forms the foundation of that context in a way that allows us to easily determine the user's awareness and maintain the continuity of that awareness. Typically, a user's presence point represents her state in some context element as sampled by sensors such as GPS, or gathered from available data such as that provided by a social

networking service's API. But as we argued in Chapter 3, a user's sampled state only gains meaning, only becomes her context, when it is considered in relation to the states of other users or objects populating the application, particularly we consider a user's context to be her nearness to those other users or objects, for some mutually negotiable measure of nearness. We shall define the user's *Aura*, the set of all points lying within a specified distance of her location, as setting this standard for her particular circumstance. It represents an estimation of the scale upon which she intends to act in the application's environment – it places a limit on the extent at which others can be considered near, or in proximity, to her. When an aura is specified for a user as a scalar value, her context can then be thought of as the users or objects that are near enough to her for their own auras to interact with hers. It is precisely these interactions, their type and value, which her awareness captures.

Our Highlight example above features a consumer, Carrie, who will gain awareness of a producer, Peggy, when her Focus overlaps in a particular manner with Peggy's Nimbus. In order to model this, we must determine the type of aura configuration that occurs between Peggy and Carrie at any given moment as their respective presences evolve. To do this, our model must be able to calculate at a minimum, two things:

1. Is a user location point inside a given aura?

   I.E. Is Carrie's location point inside Peggy's Nimbus, i.e. $C_p \in P_N$.

   And is Peggy's location point inside Carrie's Focus i.e. $P_p \epsilon C_F$.

2. Do two given aura overlap?

   I.E. Given the set of points C_F which comprise Carrie's presence (her location and the area covered by her Focus) and the set of points P_N which comprise Peggy's presence (her location and the area covered by her Nimbus) is ! ($C_F \cap P_N = \emptyset$)

Additionally, in order to determine the quantitative value of awareness as we discussed might be useful for an application like Savannah in order to increase the continuity of its users' Awareness, we must also be able to calculate:

3. The extent to which the two aura overlap

This is determined by the projection of each Aura into the other towards the User's location.

Requirements 1 and 2 give us the qualitative measure of awareness between a Consumer and Producer which allows applications to draw conclusions about the type of context that exists between them, and requirement 3 gives us the quantitative measure of the same, which allows applications to preserve continuity of context. We thus require a mathematical representation of presence in our model that facilitates the computational assessment of these three qualities both within each of the types of space – Symmetric, Asymmetric, Discrete and Continuous – that our model creates from sensed data representations, and across their various pairings.

The one property universal to the spaces our model constructs is the distance metric. The first requirement of our distance metric is that it maps every pair of points in a space's pointset to a positive value in $\mathbb{R}$ (we allow the value ∞ for pairs of points whose distance is yet to be defined e.g. as in modelling dynamic representations). Since a distance metric must be defined on every pair of points in a Pointset we can always guarantee that a definition of aura that relies on a simple scalar distance value will be able to give a definitive answer to the first of our awareness calculation questions: Is point x within Aura A? If Aura is visualised as an area centred on the user's location, we can also determine if two circles in symmetric space overlap by comparing the sum of their radii to the distance between their centres, which satisfies the second requirement, and there exist various ways to quantitate this overlap, which satisfies the requirement for calculation of awareness level. This approach works in both Continuous and Discrete spaces and since aura is reduced to a scalar value, projecting it from one space into another is simply a matter of adjusting the size of the scalar between the two spaces so as to avoid incongruent changes to presence. With aura as a scalar value representing a "circle" around the user's point of presence, calculation of awareness type reduces to a simple set of equivalence equations.

Calculation of Qualitative Awareness in a single space

Given a Consumer with presence Point L_c and Focus of radius F, and a Producer with presence Point L_p and Nimbus of radius N, type of awareness is determined in Table 544.

Type of Qualitative Awareness (AQl)	Aura Configuration	Equivalence equation
None	*The producer and consumer's aura have no overlap*	$d(L_C, L_P) > N + F$
Low	*The producer and consumer's aura overlap but the producer's point of presence does not lie inside the consumer's aura nor the consumer's point of presence inside the producer's aura*	$d(L_C, L_P) \leq N + F$ $d(L_P, L_C) > N$ $d(L_C, L_P) > F$
Passive	*The producer's point of presence does not lie inside the consumer's aura, but the consumer's point of presence lies inside the producer's*	$d(L_P, L_C) \leq N$ $d(L_C, L_P) > F$

Active	*The producer's point of presence lies inside the consumer's aura, but the consumer's point of presence does not lie inside the producer's*	$d(L_C, L_P) \leq F$ $d(L_P, L_C) > N$
High	*The producer's point of presence lies inside the consumer's aura and the consumer's point of presence lies inside the producer's aura.*	$d(L_C, L_P) \leq F$ $d(L_P, L_C) \leq N$

Table 5 – Calculating the Quantitative Value of Awareness between two Native Points of presence

We take the Aura to be a closed, rather than open, region because we feel it better reflects the way a user might expect Aura to behave in a discrete space. For example, in a discrete space if a user sets her Aura to be 6 hops, we would expect an object at a node 6 hops from her to fall within her Aura. If her Aura were an open region it would be equivalent to saying the aura stretches all the way up to but not including the node at the 6^{th} hop. This also allows users located at the same point in a space to maintain awareness of each other even when their Aura are zero without introducing extra rules. Aura are treated as non-directional because not all of the spaces we reason across support direction as a property.

In those spaces whose distance metrics are Asymmetric, we cannot simply choose one distance or the other—from Producer to Consumer or vice versa—when evaluating an aura configuration for low or no awareness. Instead we use proportional values, thus if the sum of the ratios of each aura to its respective distance is not less than 1, we reason that the aura must intersect. Formally, when the distance metric is asymmetric, the existence of low awareness is given by satisfying the inequalities:

$$\frac{F}{d(L_C, L_P)} + \frac{N}{d(L_P, L_C)} \geq 1$$
$$d(L_P, L_C) > N$$
$$d(L_C, L_P) > F$$

As we shall discuss in the next section, these awareness types not only describe the nature of an interaction between a Producer and a Consumer presence, they also inform our calculation of awareness level.

Calculation of Quantitative Awareness in a single space

Motivating Awareness as a function of Scalars

Our requirements for calculation of awareness specify that we must be able to determine the extent to which two aura overlap. For example, to appropriately size each noticeboard's overlay on the user's display in an augmented reality noticeboard application, or so that we can order and filter the notifications of nearby users according to physical and social proximity to the user receiving them in a location-aware socialising application such as Highlight. In these situations, we might imagine we are calculating the extent to which a consumer's area of interest intersects with a producer's area of influence, or the amount of the consumer's interest which is taken up by the producer. This implies that awareness level should be calculated as a ratio of the area of overlap between two aura to the area of the aura themselves, especially in the situations where we visualise a user's Aura as a circle centred on her location. However, whilst the calculation of this overlap is simple in some spaces, it can be extremely difficult in others. As a tractable example, in a Cartesian space, the overlap between two circles can be found by identifying the two points at which the circles intersect and taking the sum of the segment of each circle created by the line between each intersection point [Figure 1111104] However, if the same space were given an Asymmetric Metric we would experience problems determining how to calculate the size of those segments.

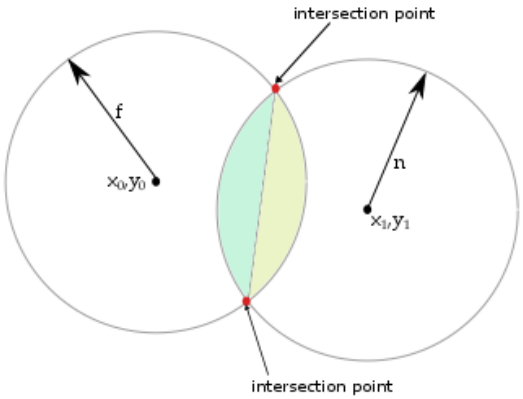

Figure 11 - Finding the coordinates of the two intersection points allows us to compute the size of the overlap (shaded areas)

Given a Consumer A and a Producer B with locations $A_L = (x_0, y_0)$and $B_L = (x_1, y_1)$ respectively, we can find the points at which the borders of their Focus and Nimbus intersect with the equations for the x and y coordinates of each intersection point given by

$$x_\cap = \frac{x_0 + x_1}{2} + \frac{f^2 + n^2}{2D^2}(x_1 - x_0) \pm 2A\frac{y_1 - y_0}{D^2}$$

And

$$y_\cap = \frac{y_0 + y_1}{2} + \frac{f^2 + n^2}{2D^2}(y_1 - y_0) \mp 2A\frac{x_1 - x_0}{D^2}$$

Where f is the size of A's focus, n is the size of B's nimbus, D is the distance between A and B's locations and A is the area of a triangle whose edges are n, f and D:

$$A = \frac{1}{4}\sqrt{(D + n + f)(D + f - n)(D - f + n)(-D + f + n)}$$

In an Asymmetric space the distance metric produces two values for D, one for the distance travelling from A to B and a second for the distance travelling from B to A, and the equations mix values that are related to travelling from A to B – the focus—with values that are related to travelling from B to at—the nimbus. This makes it impossible to pick which distance value to use in a particular part of the equations. Indeed, in a continuous space whose distance metric is non-Euclidean, the calculation of areas requires an understanding of

the implementation details of a metric that our model is trying to abstract away from. Hyperbolic geometries, for example, require knowledge of the plane's Gaussian curvature in order to compute the area of a disk. We feel a thorough investigation of the sorts of geometries that might arise when Continuous context representations are treated as spaces and the details of the quantitative intersectional relations they might support is outside the scope of this work. Our goal instead is to investigate and implement a simple method of reasoning about context and awareness that can be applied universally to many types of space. This difficulty reasoning about area, in Continuous Asymmetric spaces in particular and non-Cartesian Continuous spaces in general, is what drives our adoption the scalar approach to awareness calculation, taking awareness level as a function of the depth to which one aura projects into another. This also brings our interpretation of awareness level into line with our reading of context as constructed primarily by proximity between users.

Maxing Awareness at User Locations

When constructing awareness level, we are faced with the choice to allow awareness to be unbounded, to increase potentially infinitely given the correct circumstances, or to place bounds upon it. An example of unbounded awareness level can be imagined in a location-aware dining scenario. Restaurants use awareness level to trigger the display of promotional materials including menus, e-coupons and virtual advertising billboards in order to encourage users to dine with them. The greater a consumer's awareness of a restaurant, the larger their promotional materials will appear as overlays on the user's view of their environment. A consumer, Carrie, is located 100 metres from a restaurant with a nimbus of 50 metres. As Carrie expands her focus to greater than 50 metres, she gains low awareness of the restaurant, and begins to see its billboard on her mobile devices augmented view of the environment. As Carrie increases her focus size, her awareness level, the projection of her focus into the restaurant's nimbus grows and she sees the billboard similarly increase in size on her display. When her focus reaches 100 metres, she gains Active awareness of the restaurant which triggers the display of an e-coupon. If Carrie continues to increase the size of her focus without changing her location, her awareness type will remain unchanged, but if her awareness level is not

bounded in some way, her awareness of the restaurant will continue to increase. If Carrie expands her focus by orders of magnitude, to 6km for example, the restaurant's promotional materials overlaid on her view of the environment will increase in size, similarly by orders of magnitude, and may increase to an extent that they block Carrie from seeing anything else in her view, despite the fact that her increased focus indicates she is interested in seeing information from restaurants further afield. Although this problem can be attacked at an application level by allowing developers to restrict the maximum size of an overlay, our approach to making context more continuous is to provide a system of support for users, such as those in Savannah, through the signalling of the existence of objects outside their direct context, i.e. which are located outside their focus or whose nimbus they are located outside of, in order to make the effects of navigation and manipulation of the system clearer. This goal suggests High Awareness as an upper bound on total quantitative awareness just as No Awareness is the lower bound, and Active and Passive Awareness as maximum bounds on Focus and Nimbus projection respectively. An approach to calculating awareness level which steps outside of this easing-in of objects to user context is at odds with our model's aims, so we continue to explore the calculation of awareness as a value whose bounds are set as conditions of High, Active and Passive awareness. That is, given a Consumer with location L_C and focus F, and a Producer with location L_P and nimbus N, when the L_C lies within N, the value of projection of the Nimbus into the Focus is bounded at a value of 1, likewise for the projection of the Focus into the Nimbus, when L_P lies within F. We calculate projection by taking the scalar size of overlap between the two aura as a proportion of the scalar size of each aura. A scalar overlap is preferred to a volume or area because, as mentioned previously, some spaces will not support the calculation of area or volume. For example, a Continuous space may use a metric which determines distance between points based on travel time between them. As this generates a piecemeal Non-Euclidean metric, it cannot be used to determine area or volume. There are, naturally, many spaces within which we might be able to calculate awareness as the area or volume of overlap between two Aura, however, this raises the question of how we should reconcile these awareness

values with those calculated in spaces where area is not supported when reasoning across two spaces. For this reason, we choose a scalar measure of overlap which is, by the existence of the distance metric, necessarily supported in and across all spaces.

Assuming the distance between the Consumer and Producer as $\mathrm{d}(L_P, L_C) = \mathrm{D}$ awareness is not High or Active, we calculate the projection of the Focus into the Nimbus, the consumer's portion of the qualitative awareness (AQt_C) as

$$AQt_C = \frac{(N + F) - D}{N}$$

When awareness is not High or Passive, we calculate the projection of the Nimbus into the Focus, the Producers portion of the qualitative Awareness (AQt_P) as:

$$AQt_P = \frac{(N + F) - D}{F}$$

To calculate the total awareness level (AQt_T), we take the mean of AQt_C and AQt_P, however our model can still expose both AQt_c and AQt_P to applications to allow a more detailed assessment of awareness. Under the five awareness types, our full calculations for Awareness are detailed in Table 655.

A Working Example

If our model of Highlight uses a continuous symmetric space, representing the GPS location system, to position users, we can illustrate the calculation of awareness type and level with the following example. Let consumer Carrie's location be (52.9600, 1.1233) and producer Peggy's be (52.9500, 1.1333), then the Euclidean distance between them is approximately 0.0141. If Carrie's Focus is 0.02 and Peggy's Nimbus is 0.01 then we can deduce that Carrie has Active awareness of Peggy because:

$$(D < F) \wedge (D > N)$$

We can then calculate the level of Carrie's awareness of Peggy as

$$\frac{1+\frac{(0.01+0.02)-0.0141\ldots}{0.02}}{2}=0.8964$$

Qualitative Awareness (AQl)	Calculations performed to find Quantitative Awareness (AQt)
None	$AQt = 0$
Low	$AQt_P = \frac{(N+F)-D}{F}$ $AQt_C = \frac{(N+F)-D}{N}$ $AQt_T = \frac{AQt_p + AQt_C}{2}$
Passive awareness	$AQt_P = 1$ $AQt_C = \frac{(N+F)-D}{N}$ $AQt_T = \frac{AQt_p + AQt_C}{2}$
Active awareness	$AQt_P = \frac{(N+F)-D}{F}$ $AQt_C = 1$ $AQt_T = \frac{AQt_p + AQt_C}{2}$
High awareness	$AQt_P = 1$ $AQt_C = 1$ $AQt_T = \frac{AQt_p + AQt_C}{2} = 1$

Table 6 – Calculating Quantitative Awareness under different Qualitative Awareness Values

In our model we treat Aura as extending out from a user's location to cover all points permitted by the space's structure (if one is present) and its size. Since

an asymmetric space's distance metric may produce different values for the distance between a Producer and Consumer as measured from the Producer or from the Consumer, we must adjust our projection calculations in Asymmetric space to take account of this. To do this we stretch the Aura which we are measuring projection into in order to make it comparable to the aura projecting into it. Taking the distance from Producer to Consumer as D_{PC} and the distance from Consumer to Producer as D_{CP}, we can rewrite the equation for projection of Focus into Nimbus as.

$$AQt_C = \frac{N\frac{D_{CP}}{D_{PC}} + F - D_{CP}}{F}$$

And projection of Nimbus into Focus as:

$$AQt_P = \frac{N + F\frac{D_{PC}}{D_{CP}} - D_{PC}}{N}$$

For example, modelling a location-aware e-coupons application, our distance metric might take into account the time required to travel to a venue. Given the distance from Peggy to Carrie as 0.015 and from Carrie to Peggy as 0.03. If Carrie's Focus is 0.02 and Peggy's Nimbus is 0.01, we can determine that in this case, Carrie has Low awareness of Peggy since:

$$(D_{CP} > F) \wedge (D_{CP} > N) \wedge \left(\frac{F}{D_{CP}} + \frac{N}{D_{PC}} \geq 1\right)$$

i.e.

$$(0.03 > 0.02) \wedge (0.015 > 0.01) \wedge \left(\frac{0.02}{0.03} + \frac{0.01}{0.015} = 0.\dot{6} + 0.\dot{6} \approx 1.33 \geq 1\right)$$

Given this, we can calculate Carrie's awareness of Peggy as

$$\left(\frac{N\frac{D_{CP}}{D_{PC}} + F - D_{CP}}{F} + \frac{N + F\frac{D_{PC}}{D_{CP}} - D_{PC}}{N} \right) \div 2$$

$$= \left(\frac{0.01\frac{0.03}{0.015} + 0.02 - 0.03}{0.02} + \frac{0.01 + 0.02\frac{0.015}{0.03} - 0.015}{0.01} \right) \div 2$$

$$= (0.5 + 0.5) \div 2$$

$$= 0.5$$

Awareness Challenges in Asymmetric Discrete Spaces with Edge sets

The previous section explored our model's processes and equations for calculating awareness type and value, and in doing this we discussed some of the restrictions that arise from the need to support awareness calculation in each of our four different types of space. Although our calculations work in the majority of situations, the calculation of awareness in some spaces poses problems particular to their structure and we discuss these now.

Assume that the directed subgraph illustrated in Figure 1212115 is part of a larger graph which models a discrete asymmetric space with an Edge set $E \subseteq V \times V$. Given a consumer located at node A with focus of size 5 and a Producer located a node B with a nimbus of size 7, the space's distance metric gives the distance from Producer to Consumer as 16 and the distance from Consumer to Producer as 13. From this our model determines that the awareness that exists between them is not High, Active or Passive. Using the asymmetric awareness calculation tells us that the awareness between them is none:

$$\frac{F}{D_{CP}} + \frac{N}{D_{PC}} < 1$$

$$\frac{5}{13} + \frac{7}{16} = 0.822 < 1$$

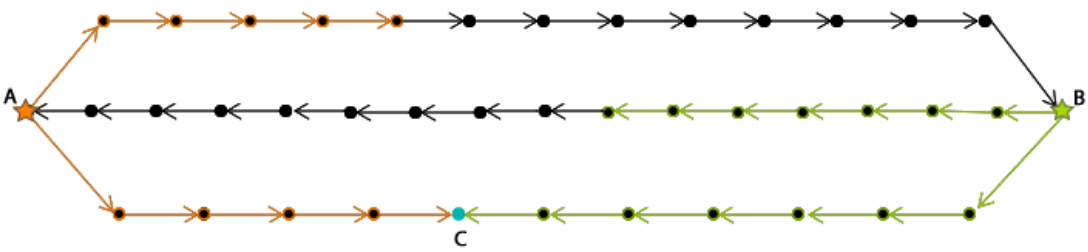

Figure 12 – The aura of A and B intersect at node C but the path from A to B and the path from B to A are both longer than the sum of their aura.

However, we can see from Figure 1212115 that since each Aura spreads out from its user's location as far as the structure of the space and its size permits, in fact the Focus and Nimbus of the Consumer and Producer do overlap at the vertex labelled C.

Since some Discrete spaces are imbued with their own structure which limits the immediate neighbours of any point to a finite number of vertices, spaces of this type support the calculation and enumeration of the set of points that comprise a user's aura in a way that Continuous spaces do not. It is this that allows us to see that the two aura in the example overlap at point C. In a Symmetric Discrete space, such a situation, where the distance between a Consumer and Producer is longer than the sum of their Focus and Nimbus yet the two Aura overlap, cannot exist. To see why, consider Figure 1313126 - the symmetric version of the graph in Figure 1212115.

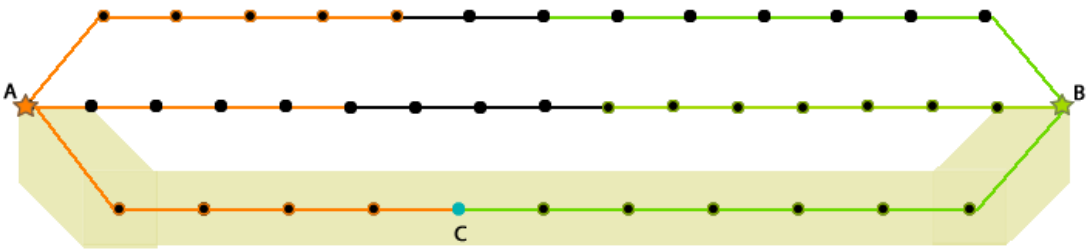

Figure 13 – When the space is symmetric the path through C becomes the shortest path

In this instance the path from A to C becomes also a path from C to A, likewise the path from B to C becomes a path from C to B, creating a path (highlighted in Figure 1313126) from point A to point B through point C of length 12. Since

this path now becomes the shortest path from A to B, the distance from A to B must now be 12 (obeisance to the triangle inequality of our model's distance metric rules) and the applying the awareness type calculation to this new value yields a result of low awareness:

$$N + F \geq D$$

$$7 + 5 \geq 12$$

This issue with Awareness in structured Asymmetric Discrete space arises in a variety of spatial configurations, whenever there can be found one or more vertices which are both less than distance F from the Consumer and less than distance N from the producer, but where the shortest path from Consumer to Producer is larger than F and the shortest path from Producer to Consumer is larger than N. The simplest way to detect the existence of such vertices is to determine if the intersection of the Focus and Nimbus sets, so allowing V_N as the set of vertices within distance N of the Producer, and V_F as those within distance F of the Consumer, in an Asymmetric Discrete space we determine a state of No Awareness between Producers and Consumers with the equations

$$D_{CP} > F$$
$$D_{PC} > N$$
$$V_N \cap V_F = \emptyset$$

And low awareness with:

$$D_{CP} > F$$
$$D_{PC} > N$$
$$V_N \cap V_F \neq \emptyset$$

An unstructured Discrete Asymmetric space, however, behaves like a Continuous Asymmetric space and does not require such operations. The determination of the existence of an intersection between the two aura in a structured Discrete Asymmetric space can be achieved by first constructing greedily and breadth-first a tree T_F of depth F rooted at the Consumer's location and a hashtable H_F which stores records of the vertices in this tree,

then constructing breadth-first, a tree T_N of depth N rooted at the Producer's location where every new vertex added to T_N is also looked-up in the H_F to determine if it exists in T_F. To determine awareness type, we simply need to find one such vertex and then stop. However, if we include in each vertex's entry in H_F the cost of reaching that vertex in T_F, (such that for unweighted graphs the cost of traversing an edge is 1) and after having found the first shared vertex, we continue to build T_N comparing each vertex added to those in H_F. We can calculate the deepest projection of Focus into the Nimbus and vice versa by finding the vertex in the intersection whose cost to reach from the Consumer is lowest and the vertex in the intersection whose cost to reach from the Producer is lowest. We do this by first recording the depth in T_N of the first vertex to be found present in both trees. Since we are constructing T_N breadth first and greedily, this gives us the vertex in the intersection of Focus and Nimbus which is closest to the Producer, and its depth is that of the projection of Focus into Nimbus. Secondly we record the depth in T_F of this vertex and replace it with the depth of any subsequent vertex found to exist in both T_F and T_N whose depth in T_F is less than the currently recorded depth. Once T_N is fully constructed we will have enumerated through every vertex in the intersection between Focus and Nimbus and since we used the process to record the depth of the vertex closest to the Consumer, we have the depth of the projection of Nimbus into Focus. The same approach can be used to obtain projection of the Focus into the Nimbus under Passive awareness and projection of the Nimbus into the Focus under active awareness in order to calculate the awareness level for these configurations in structured Asymmetric Discrete Space.

Projecting a user's presence from one space into another

Thus far in this Chapter we have described the mathematical foundations and structures of presence and our methodologies for using these to calculate awareness type and level within each of the different classes of space that our model supports. Our aim, however, is for our model to support the calculation of awareness across those spaces, allowing developers to combine different representations of context. We go about achieving this cross-space awareness

calculation by first projecting a user's presence from one space into another and it is this process that we now discuss.

Projection Points and their Structure

Social curation of information is the practice of filtering or emphasising the notifications a user receives based on observed information about their friends, the more of a user's friends who show interest in a topic, the more likely it is for notifications concerning it to be presented to the user. Applications such as Highlight provide location- and context-aware socialising support to users by delivering to them notifications of others nearby with whom they share friends or interests. Whilst Facebook provides robust APIs for mapping social connections, it provides less support for mapping shared interests across its userbase. Other social networks which are more geared towards content production and consumption, such as blogging platforms, are more fertile ground for mining user interest. Imagine a Facebook and Highlight user, Cathy, has a friend Bridget who also uses Tumblr [TUMBLR, '16] (a blogging platform with asymmetric social connections, similar in structure to twitter), and since Cathy and Bridget are friends it is reasonable for an application developer to assume they share some of the same interests, that although Cathy is not a Tumblr user herself, bloggers on Tumblr who are interesting to Bridget may also be interesting to Cathy. With this in mind a space modelling the connections between Tumblr users can be built with Bridget's presence in both the space of Facebook users and the space of Tumblr users allowing the creation of a connection between the two spaces allowing the model to see those Tumblr users followed by Bridget as also being in proximity to Cathy. In chapter 3 we explained that our mechanism of joining spaces is the use of pairs of points that signify the same point in some aggregate space, which we call projection points. Here Bridget's presence on both Facebook and Tumblr creates a link between her locations in the spaces which model each social network Figure 1414137

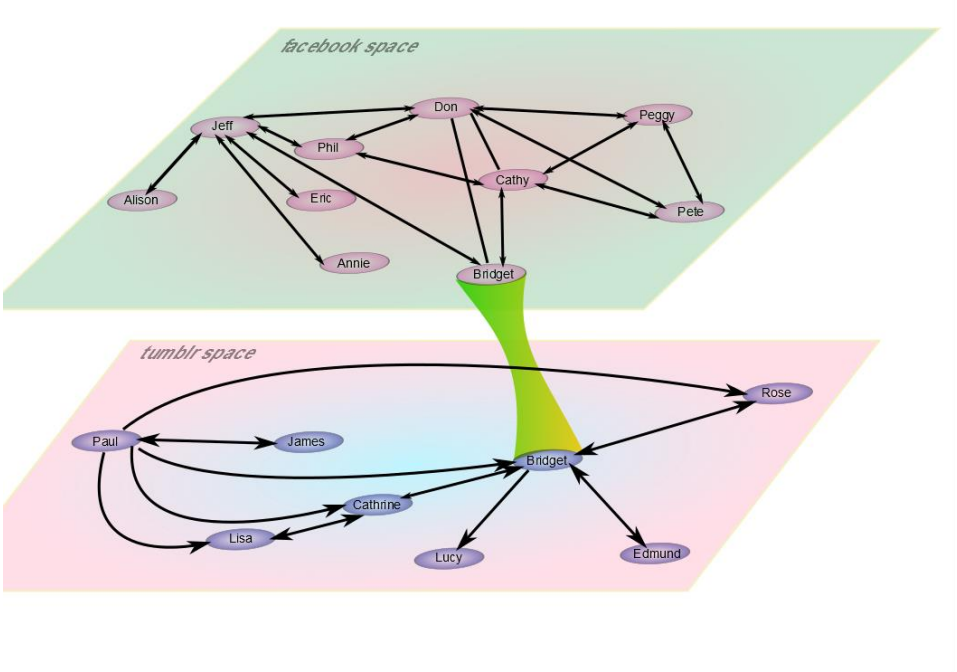

Figure 14 – Bridget's presence in both the Facebook and Tumblr spaces creates a link between the two spaces

In our model, when a user's aura in one space is incident upon a projection point, an image of the user's presence is projected into the space that the projection point links to. In our example, Cathy's aura, being incident upon Bridget's account in the Facebook space would cause an image of Cathy's presence to be created in the Tumblr space allowing her to gather notifications about Tumblr users connected to Bridget. The projection point in the second space, the Tumblr space in our example, forms the image's presence point, so Cathy would be located at the vertex in the Tumblr graph which represents Bridget's Tumblr account, and the image is also given an aura whose size is dependent on the distance of the projection point from the user's location, since Bridget and Cathy are immediate friends on Facebook, the aura of Cathy's image-presence would be only one unit smaller than her aura in the Facebook space [Figure 1515148]

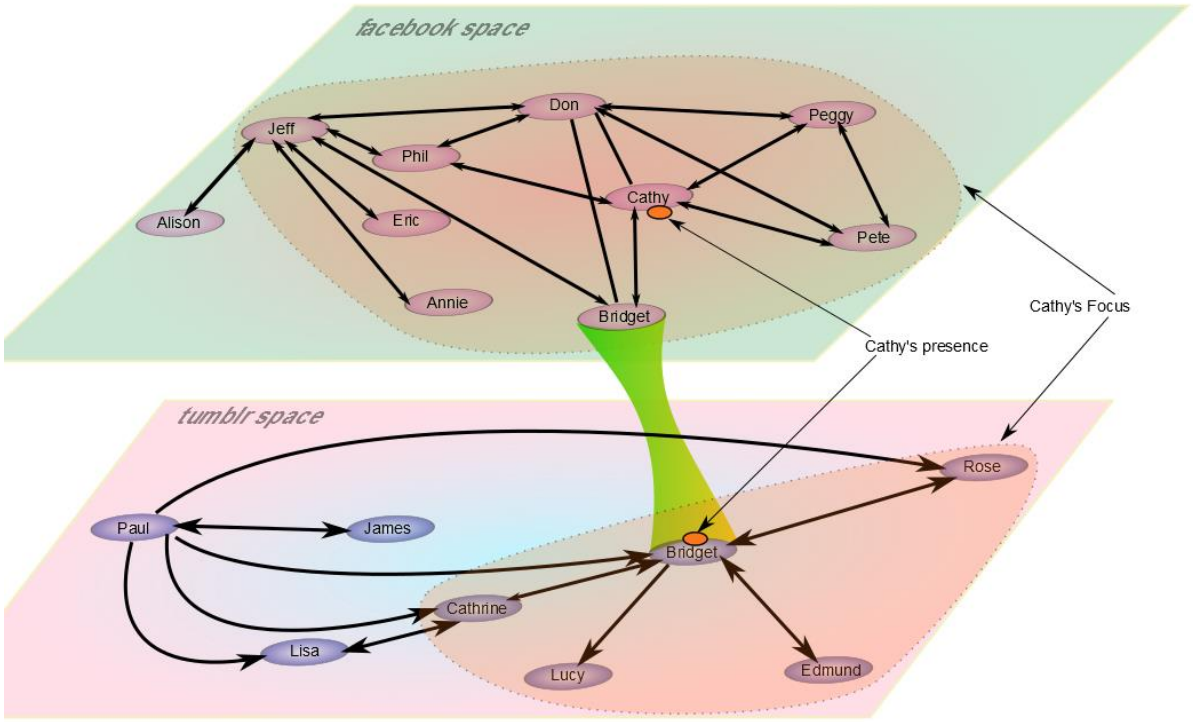

Figure 15 – Cathy can gain presence in the Tumblr space projected onto the social location of her friend's Tumblr

Since Cathy's native presence point represents her context as sampled in the Facebook space, and her aura the scale upon which she acts in that particular space, the image's point of presence represents an approximation of the value her context would take if it were sampled in the Tumblr space, given her current sampled state. The assumption, again, is that being friends with Bridget indicates they have similar interests and were Cathy a Tumblr user she would follow a similar set of blogs to Bridget.

The example demonstrates a link between two spaces where the projection point is a pair of single points, one in each space. However, representations may by radically different in structure and scale and may indeed be representations of different context elements, for example a developer may wish to link user nodes in a Facebook social connections graph to the GPS location of each user's home. These differences in structure and scale between

context representations often mean that a point in one representation maps to a set of points in another. Attempting to provide users who only have access to Wi-Fi location sampling awareness of objects that only use GPS to report their location, we might find that a single vertex in the space modelling SSID sets covers tens or hundreds of metres in the GPS space. If projection point relationships between spaces were limited to pairs of single points, we would be forced to choose one GPS point out of the set which a particular SSID set covers to represent that Wi-Fi space's vertex in the GPS space. An object with a small nimbus then, might be unable to gain presence in the Wi-Fi space, despite being located in the area covered by that particular SSID set because it is too far from the selected GPS point for its nimbus to be incident upon it. More generally, when mapping a discrete space to a continuous space, we find that a single vertex in the discrete space must often be linked to a set of vertices in the continuous space. This presents some design dilemmas for our model which we now discuss.

Contiguous vs. Non-contiguous projection points

The SPACES model is designed to support developers who work with context representations that are not fully known during development or at runtime, or which continue to evolve during runtime. The information required to build relationships between these spaces in our model, i.e. To detect and create projection points, is gathered either separately from other services, or as an adjunct to the application itself during runtime. To enable this, our model allows both the creation of new projection points during runtime and the updating of existing projection points. Since this approach supports the use of partial mappings between spaces, and as discussed in the previous section, a point in one space may map to multiple points in another, it is possible for a single point in one space to map to a set of non-contiguous points in another space. Building relationships between a Wi-Fi location space and a GPS location space can lead to situations where only parts of an area covered by a particular set of SSIDs are mapped to GPS coordinates [Figure 1616159]. When this occurs, there is a question of how to construct the proxy presence of a user projected onto those points from another space. A user's proxy could

either take a presence that encapsulates all the discontiguous points, selects one point in particular, or clones itself across the separate points, giving the user multiple presences in the space they have been projected into.

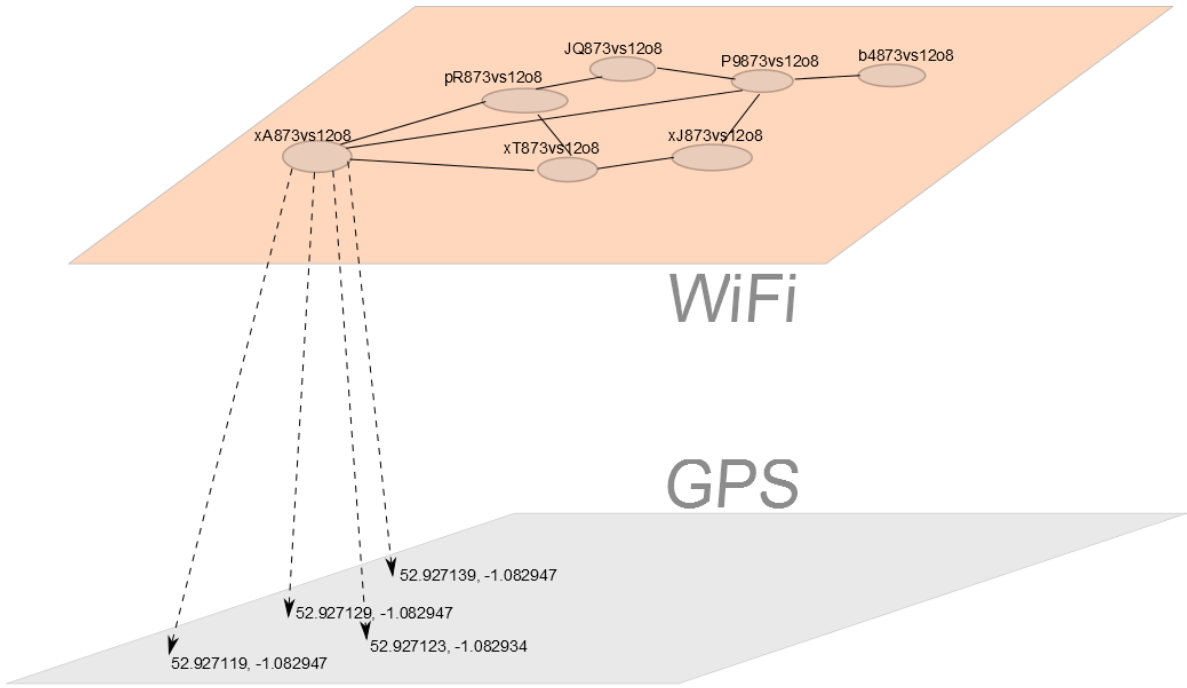

Figure 16 – The model has only enough information to link the Wi-Fi space node to a set of discontiguous GPS points

Our model approaches the calculation of awareness between a pair of users across two spaces by first calculating the type and value of awareness between the two users within each space and then aggregating those results to assign a global awareness type and value to the pair. Although there are undoubtedly situations where it would be useful to provide a user with multiple presences in a single space, where awareness of other users is then aggregated across those presences, we feel this is outside the scope of our current work, so we choose to discard the possibility of multiplying a user's presence when it is translated through a projection point onto discontiguous sets of points. We consider rather, for our current model, a solution to the projection of presence from discrete to continuous spaces that maintains at most one proxy per user per space. When mapping from a single point in one space to a set of points in

another, it may be possible to use properties such as the number of points in the mapped-to set or the density of the set – perhaps given by the average distance between each pairs of points – to give a measure of strength of the projection between the spaces, however this is something we felt was outside the scope of our initial model and did not explore.

The Form of Projected presence

Since a proxy takes as its point of presence the projection point paired with the one which the user's aura was incident upon, this design choice raises the question of how to assign a presence point to a proxy when the projection point's pair is not a point, but a set of points, either contiguous or non-contiguous. Recall that we chose our definition of presence and aura very specifically to support the three requirements we gave for calculating awareness type and level:

1. Determining if a given presence point lies inside an aura
2. Determining if a two aura overlap
3. Determining the extent to which two aura overlap.

If we wish to allow a user's proxy to define its point of presence as either the set of points equal to the contiguous set used in the projection point pair, or some approximation of that non-contiguous set made contiguous, we must change the first requirement for calculation of awareness to:

1. Determining if a given *set of points* lies inside an aura.

This is not so simple as determining if a single point lies inside an aura. In a Discrete Space we could enumerate over the points in the Aura and compare them to the given set of points, but in Continuous space, to point set being some subset of $\mathbb{R}^*$, the cardinality of the Aura set and the given set of point is infinity. As covered in our discussion about the motivation of awareness as a function of scalar values, under some distance metrics it can be difficult or impossible to calculate area overlaps, and this includes calculating the containment of one set within another. Furthermore, even if we approximate the set of points to a more regular shape and structure, for example wrapping it

in the same centre-point and scalar structure as a user's presence takes, we can still see issues with awareness arise.

Let the image-presence's set of points of presence be P', the centre point of the approximation of this set be C' and the radius of the set be R'. Then we can determine whether P' is contained within a user's aura by evaluating whether the size of the aura, A, is greater than the sum of R' and the distance from L, the user's location, to C'. That is:

$$A - R' + d(C', L) \geq 0$$

If this is the case, the set of points of presence is entirely contained within the Aura and the awareness type may be High, Active or Passive dependent on rest of the aura configuration and the intents of the users. If A is smaller than d(C', L) minus R', the image's set of location points is entirely outside the Aura and awareness may be low or none. However, if A is not more than R' smaller than d(C', L) the Aura *partially* overlaps the image's location creating a state of awareness between Low and Active/Passive/High. In chapter 3 we discussed how developers might assign semantic meanings to our five types of awareness, for example a location-aware strategy gamer with low awareness of a fellow player would be able to see some of the structures the player had built in the game, but not interact with the player directly; or a tourist with Active awareness of a monument would be able to access detailed information about it as they were actively indicating their interest in it. The semantic meaning of a state between Low and Active/Passive/High awareness is unclear and although we could choose to amalgamate it with one type of configuration or another, in order to maintain simplicity in our model, we choose to restrict a point of presence to a single point. To do this for a proxy when a projection point pair involves a set of points, we again calculate C', the centre of the set of points, but in this approach we take that single point as the proxy's point of presence. This allows us to maintain use of the awareness type and level calculations that we described earlier in the chapter, without changing the semantic meaning of the proxy which is an approximation of the context a user would have, given their current context, if they possessed the technology to be directly present in the space themselves. This also gives us the opportunity in our SPACES

platform to allow developers flexibility in how they implement the bounding box structures involved in projection point relationships, providing that they offer a method to access the centre point of the box.

Scale and Projected presence

Our reading of context as proximity places distance and scale at the centre of our model, however when developers build representations of context elements, the scales they employ are oftentimes unrelated to those of other representations. In order to reason about context across different representations our model must be able to translate between these scales.

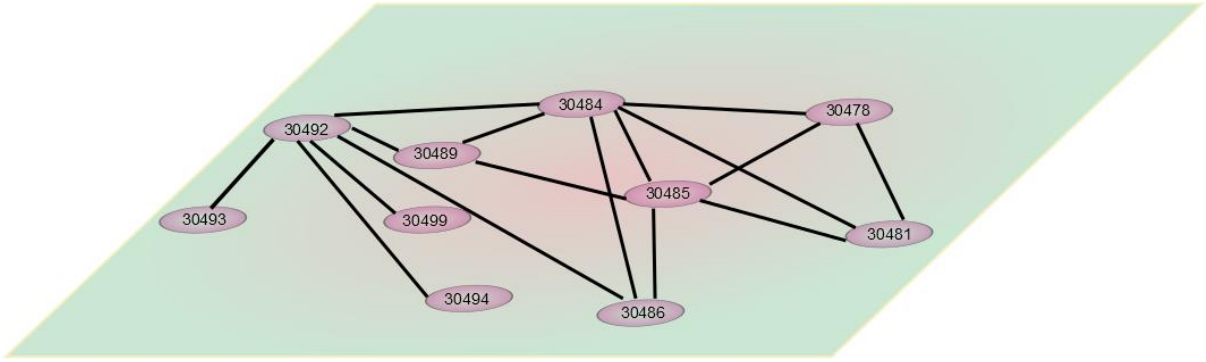

Figure 17 – An example of a GSM space

For example imagine Carrie is using GSM sensing technology to determine her location [Figure 17171610], whereas Peggy is using Wi-Fi. illustrates the space that Carrie's location technology places her in and we can see from this that the edges between vertices have no weights, or an implicit weight of 1. The space that Peggy's location technology has placed her in is shown in [Figure 18181711] and we can see that edges between vertices carry a variety of weights. Imagine Carrie is located at a projection point, allowing her to gain a proxy in the Wi-Fi space. With her aura designed to extend over a space where the edge weighting is just 1, the aura of Carrie's proxy would only cover the vertex where she is located. In order for her proxy to better represent Carrie's actual presence, we must manipulate her projected aura to reconcile the difference in scale between the spaces. Our model is designed to support developers in several scenarios: Those who have userbases with access to

heterogeneous sensing technologies who are thus using multiple representations of context, those who wish to incorporate context information gathered by other applications or share the context they gather with other applications, and those who are building applications on representations that are evolving over time. In each of these cases, our aim is to provide a system that removes from developers the burden of mapping either the inner workings of every representation they incorporate into their applications, or managing the relationships between those representations. Since our model does this by abstracting context representations—dealing for example in distance metrics which output scalar values on the input of pairs of points, rather than the mechanics of calculating distance itself, which is left to developers who are familiar with the particular representation— we do not attempt to reason about differences in scale between spaces through the model, rather we provide a mechanism for developers to specify the ways in which to manipulate any aura as it is projected out of, or projected into a representation space. The result is that developers can manipulate an aura constructed in their representation space to a neutral, or pre-agreed upon scale, and from that neutral, or pre-agreed scale, to the scale of their representation.

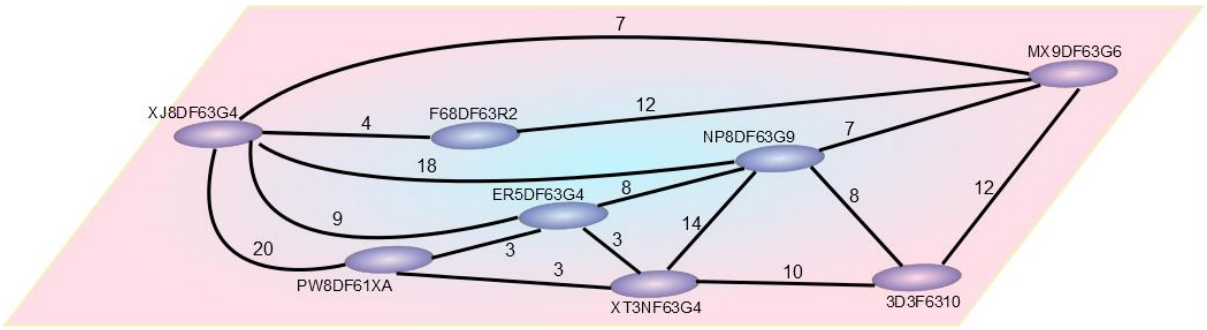

Figure 18 – An example of a Wi-Fi space

Thus, given a point p in their representation, which is one half of a projection point pair, and a scalar value, which is the aura a to be projected into another space, the developer should provide a function $O(a,p) = a'$ which scales the aura to a neutral value. And similarly, given a point p' which is the other half of the projection point pair, and a scale value a' which is the neutrally scaled aura to be projected into the representation, the developer should provide a function $I(a',p') = a''$ which scales the aura from a neutral value to one

appropriate to the space. We require the function to take the projection point as one of its inputs since scale can vary locally across some of the spaces we encounter. For example, in the GSM space where edges were unweighted, a single transition between vertices that represent cells in a rural area might represent a distance of tens of miles, whereas a transition between vertices that represent cells in an urban centre would more likely only represent a distance of tens of metres. An application which understands this and manipulates user aura within a GSM space in response to changes in the space's local scale should be given the opportunity to reflect those local differences in scale when projecting user presence into other spaces. Applying this approach to our example, if Carrie's presence is now projected into the Wi-Fi space, the O transformation for the GSM space stretches her aura to a value of 300, and the I transformation for the Wi-Fi space squashes it to a value of 30. Now Carrie's projected presence has an aura that covers several vertices in the Wi-Fi space.

Given two or more projection point pairs, it may be possible for our model itself to determine the scale transformations required upon aura by comparing the values that each space's distance metric returns for the distances between the two pairs of points in each space. We chose to forego this approach in our initial implementation of the model, because it was felt an exploration of the results this creates given spaces with varying local scale was outside the scope of this work.

On the Accuracy of Projections

Imagine Carrie, a Highlight user, is using GPS as her location sensing technology. Her device detects she is located at [53329233,-890107], Peggy, meanwhile is using Wi-Fi to provide the application with her location, which is translated in the model to the vertex ID (pR873vs12o8) [Figure 19191812]. Initially, Carrie cannot gain awareness of Peggy because the application has no way to reconcile the two different sensing technologies. If we introduce a projection point in Carrie's vicinity at [53329230,-890114], which is paired with a point in Peggy's vicinity, at (P9873vs12o8), Carrie and Peggy may each gain a proxy presence in the other's space, allowing them to gain awareness of each other.

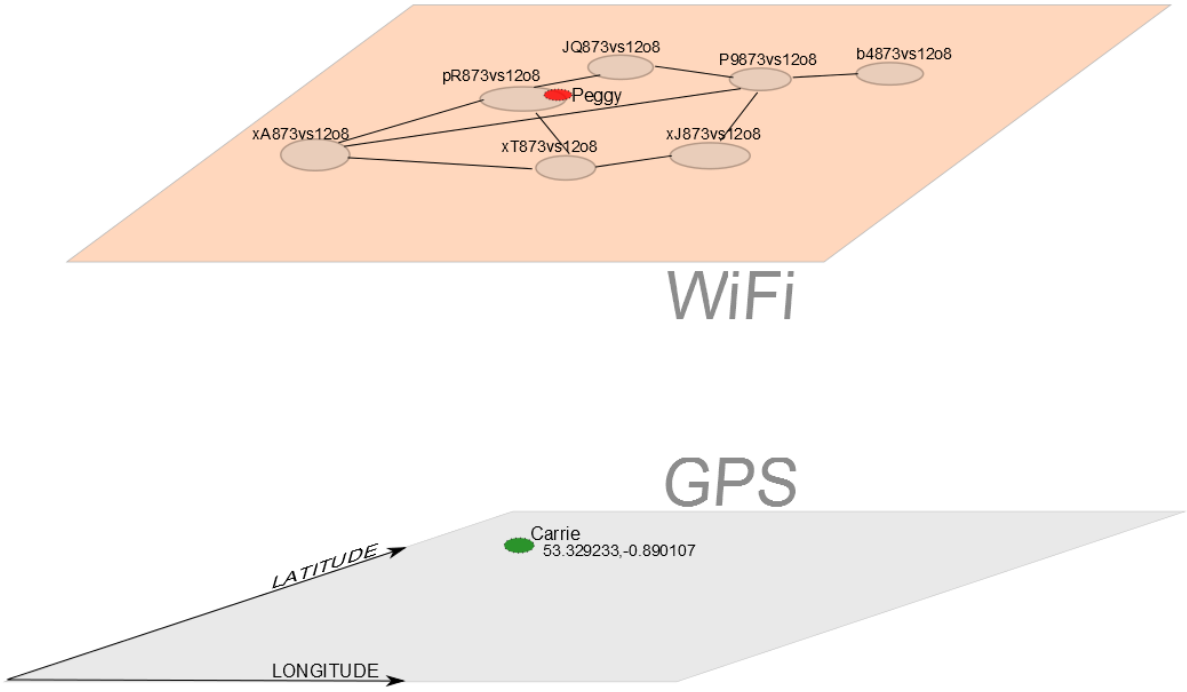

Figure 19 – Carrie located in the GPS space, Peggy located in the Wi-Fi space

Carrie's aura is 10 and the projection point is 7.6 away from her location, so when Carrie's presence is projected through the proxy point at [53329230, -890114], it appears in the Wi-Fi space as a presence point at [P9873vs12o8] with aura [2.4]. When Peggy's presence is projected through the point, it appears in the GPS space at [53329230, -890114] with aura 3, this is illustrated by [Figure 20201913].

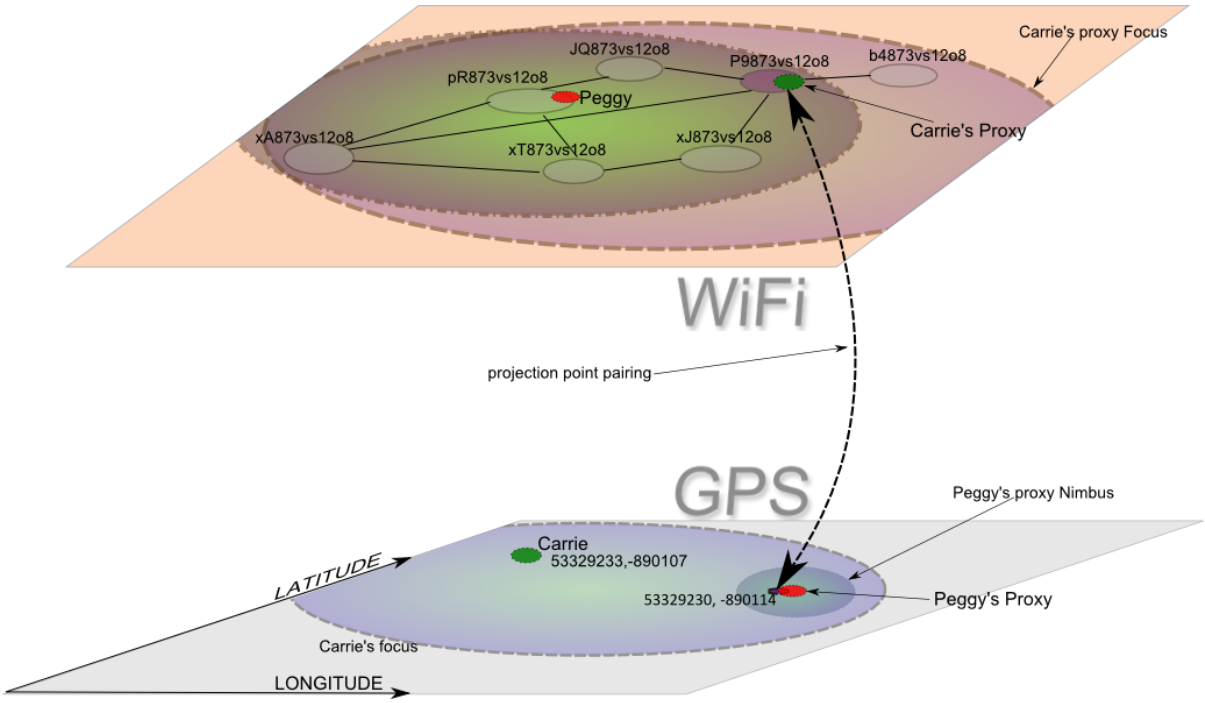

Figure 20 – Peggy gains presence in the GPS space through a projection point

Since aura is described as a scalar value, the proxy for the user's presence does not exactly mirror their presence in the second space unless the projection point happens to coincide with the user's location. We can see this in [Figure 21212014] where we imagine that we have complete knowledge of the way in which the GPS space aligns with the Wi-Fi space but have only entered one projection point pairing into the model. By this illustration, Peggy and Carrie should have high awareness of each other, however, in the model with its single projection point, Peggy's projected Nimbus in the GPS space does not reach several of the coordinates associated with vertices that her Aura covers in the Wi-Fi space.

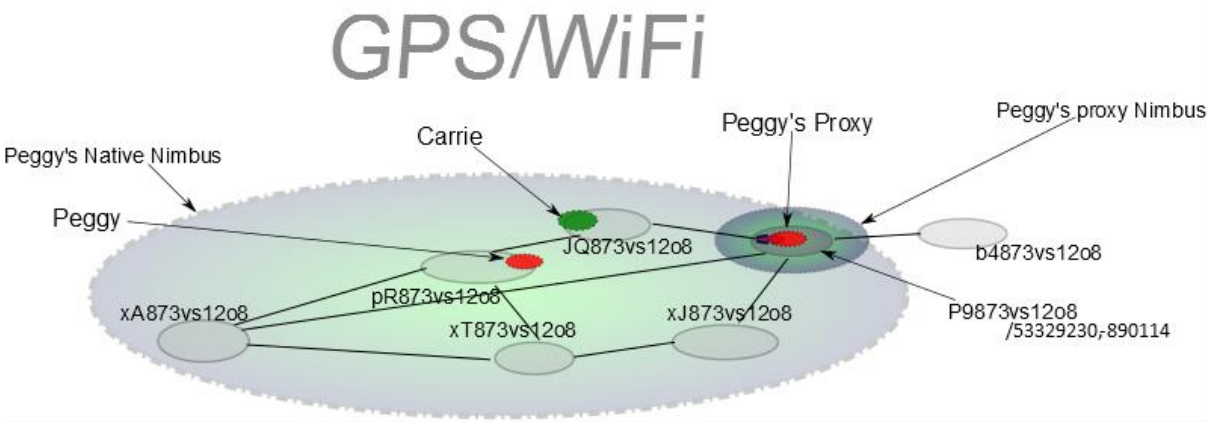

Figure 21 – How we might assume the Aggregate space looks

If we introduce a second projection point pair linking coordinate [53329225, -890107] with vertex (xA873vs12o8), and project both Carrie and Peggy's presences through that pair, Peggy's projected Aura improves its accuracy, whereas the accuracy of Carrie's projected presence degrades. Although it is apparent from [Figure 22222115] that Carrie's presence is 5 distance units North of the projection point, we cannot use this information to translate Carrie's projected presence to a more accurate position in the Wi-Fi space, because as discussed earlier, not all spaces support concepts like direction—particularly discrete spaces. Although we are able to describe the path from Carrie's location to the projection point, since we cannot orient Carrie's location relative to the projection point in the Wi-Fi space in a manner which can be translated into the GPS space, we cannot transfer that information across the spaces to be applied to Carrie's projected presence.

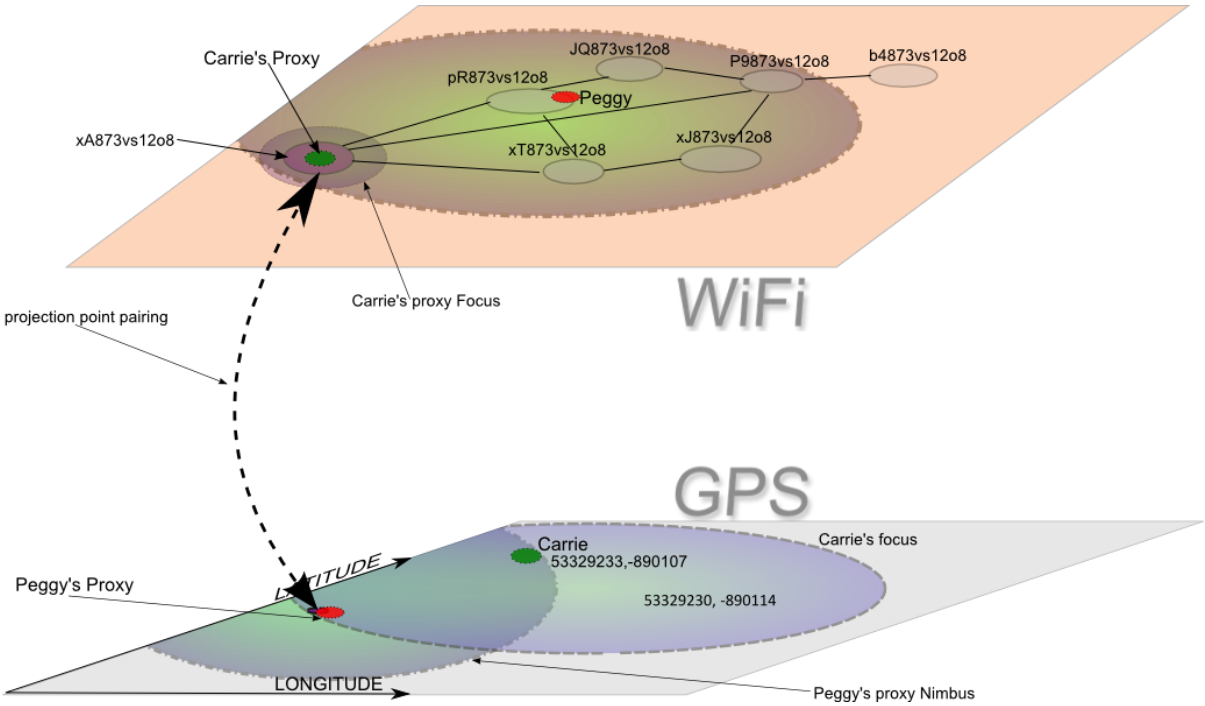

Figure 22 – Carrie gains presence in the Wi-Fi space via a projection point

As a rule, the further the projection point used is from the user's location, the further a projection accuracy in representing the user's presence in the second space decreases. Because of this inverse relationship between distance and projection similarity, in the event that there exist two or more projection points within a user's aura, our model specifies that the point closest to the user's location is used. Where multiple points are the same distance from a user's location, we pick the point which has the most sensed data supporting it. This design choice can result in a situation where a distance from a Consumer in one space to a Producer in another is asymmetric, even though the spaces themselves are symmetric, however, we argue this would imply that the currently understood relationship between the spaces is asymmetric and thus the aggregate space that they co-exist within is also asymmetric.

We discuss the possibility of creating transferable information about the relationship between user presence and presence points by analysing the presence's spatial relationship to multiple projection points in Chapter 8, our section on future work.

Increasing information about the relationships between spaces increases the accuracy of projected presences by increasing the chances that a user is located

at, or very near to, a projection point. Our model is designed to support applications that accumulate information about the relationships between spaces over time and this approach ensures that where information about the relationship between spaces is collected in the background to user activity within the application, the areas of the context space most popular with users will approach total or high levels of accuracy in presence projection more quickly than those less popular, less trafficked areas of the space. We feel this is an acceptable compromise between the need for accuracy and the need to support relationships between spaces that do not necessarily support the same semantics such as compass bearings.

Aggregating Awareness Levels across representations

Given any Consumer and Producer located in different spaces and each projected into the other's space, our model creates four aspects of aura interaction which must be reconciled in order to calculate the awareness that exists between the two users. These are:

- The projection of the Consumer's Focus into the Producer's proxy Nimbus.
- The projection of the Producer's proxy Nimbus into the Consumer's Focus.
- The projection of the Consumer's proxy Focus into the Producer's Nimbus.
- The projection of the Producer's Nimbus into the Consumer's proxy Focus.

The first two interactions occur in the Consumer's space, the second two in the Producer's. Our process for calculating a level of awareness involves calculating separately the projection of each Aura into the other and then taking the average of these two values. If we treat the two user/proxy configurations in the same way as we treat a user/user configuration the model produces four values which we can average, however, calculating the projection of the user's aura into the proxy's aura in this way does not provide us with an accurate value for strength of that aura interaction.

Imagine Peggy has a presence with Nimbus of size 0.020 in a GPS space at (52.9600, 1.1233) and gains a presence in a GSM space through a projection point pairing P_1 at (52.9600, 1.1333). Since the distance metric determines that the projection point is of distance 0.01 from Peggy's location, her Nimbus to be projected out of the GPS space is only 0.01 and this is submitted to the GPS space's output function which stretches it to a value of 1. This value is then transformed as it enters the GSM space to a value of 5, giving Peggy an image-presence in the GSM space with a nimbus of size 5. If Carrie has presence in the GSM space with an aura of 12, at a distance of 14 from Peggy's projected presence, the overlap between Carrie and Peggy's aura will be 3. Calculating the awareness level, A_C, the projection of Carrie's Focus into Peggy's image nimbus, as if Peggy were a user native to the space, gives us

$$\frac{(N+F)-D}{N} = \frac{5+12-14}{5} = 0.6$$

However, since Peggy is not located at the projection point, the model has not projected her entire Nimbus into the GSM space. In order to correctly calculate how deep into Peggy's nimbus Carrie's Focus reaches, we must account for the portion of Peggy's Nimbus which covers the distance from Peggy's native point of presence to the projection point in the GPS space. We do this by calculating the ratio between the size of the aura projected out of the GPS space and the native aura's total size.

$$\frac{0.01}{0.02} = 0.5$$

This is then used to adjust the calculated awareness level, A_{C_2}.

$$0.6 \times 0.5 = 0.3$$

Since our model already calculates the size of the portion of the aura to be projected out of the space as part of its creation of the proxy's aura, we can simply retain this information with the rest of that pertaining to the proxy. The projection of a proxy aura into a native aura describes the extent into the native aura that the proxy aura reaches as a proportion of the size of the native aura and because this value is dependent on the size of the native aura, not that of

the proxy, we do not need to adjust our original awareness level calculation in order to apply it here.

As discussed in Chapter 3, when there exists Passive, Active or High awareness between a User and a Proxy, that awareness type does not necessarily translate across to a High, Passive or Active awareness across the spaces, even if the projected user is located at the projection point in her native space. This is because a user's presence is only projected through a projection point pairing if it is the closest such pairing to her. Imagine instead that in our above example Carrie's Focus is 16, allowing it to overlap the location of Peggy's projected presence. This gives Carrie an Active awareness of Peggy's proxy presence. Since Carrie has active awareness of Peggy's proxy, the intra-space awareness level, A_{C_1}, calculated for the projection of her Focus into Peggy's proxy Nimbus, is 1. However, since Carrie's aura is not incident upon Peggy's native location, but upon her Proxy's location, the type of interspace Qualitative awareness calculated for her by the model is not necessarily Active. This difference between intra- and inter-space awareness is accounted for in Quantitative Awareness when we adjust the awareness level, calculated for the projection of Carrie's Focus into Peggy's Proxy nimbus, by reducing the value of the quantitative awareness as previously discussed.

Once we have calculated the value of interaction in each of the two spaces, those values must be combined to yield a final interspace awareness level between the two users. Although it extends the user's presence into a second space, a user's proxy is not an extension of the scale upon which the user's context acts, but a parallel expression of it in a different representation of that context. Because awareness level in our model signifies the extent to which two users are in proximity to each other, relative to the respective scales upon which each of them act, our aggregation selects the configurations which produce the greatest proximity to form the total awareness level. To do this, we pair the projection of Carrie's Focus into Peggy's proxy Nimbus with the projection of Carrie's proxy Focus into Peggy's Nimbus, which gives us values for the projection of Carrie's set of Focuses into Peggy's set of Nimbuses, and similarly the projection of Peggy's Nimbus and proxy Nimbus into Carrie's

proxy Focus and Focus, which gives us values for the projection of Peggy's Nimbuses into Carrie's Focuses. We then take the larger value of each pairing to give us the maximum occurring projection of Focus into Nimbus and vice versa and use these two values to calculate the total quantitative awareness.

Returning to our example, imagine there is a projection point pair P_2 located in the GSM space at a distance of 3 from Carrie's point of presence. Through this Carrie can gain a second presence in the GPS space. The value of her Focus as it is projected out of the space will be 9, and this is manipulated by the GSM space to a neutral value of 180. The GPS space manipulates this neutral value to create a proxy Focus of size 0.018, and as the other half of the projection point pair is the set of GPS space points with centre (52.9600, 1.1533) the distance between Carrie and Peggy is 0.03.

The awareness values in the GPS space between Carrie's proxy and Peggy are then as follows:

$$A_{C_1} = \frac{(N+F)-D}{N} = \frac{0.02+0.018-0.03}{0.02} = 0.4$$

$$A_{P_1} = \frac{(N+F)-D}{F} \times \frac{F_p}{F} = \frac{0.02+0.018-0.03}{0.018} \times \frac{9}{12} = 0.03$$

In the GSM space, we have already calculated the value of A_{C_2} to be 0.3, we now calculate

$$A_{P_2} = \frac{(N+F)-D}{F} = \frac{5+12-14}{12} = 0.25$$

Selecting the larger of each pair of values for A_{C_i} and A_{P_i} gives us a total awareness level of

$$\frac{A_C + A_P}{2} = \frac{0.4+0.25}{2} = 0.325$$

This approach can be extended to calculate an aggregated awareness between two users across three or more spaces, however, in order to preserve the simplicity of our initial model, we restrict awareness calculations to those between two users or a user and a proxy. If two users, each located in a

different space, are projected into a third representation, we do not calculate the awareness between their two presence images in that space, or aggregate that with the awareness they have of each other in their respective native spaces. We feel this is outside the scope of our initial work, but may represent an interesting avenue for future work.

Awareness across Evolving Spaces

Imagine Peggy's location in her native GPS representation is very close to a projection point pairing with the GSM space. Through this pairing, Peggy's presence will be projected into the GSM space. If Carrie, located in the GSM space at a distance of 14 from Peggy's projected presence has a Focus of size 16, she will have Active awareness of Peggy's projected presence.

However, if there exists another projection point pairing in the GSM space, located very close Carrie, the model will use that point to project Carrie's presence into the GPS space. If this pairing projects Carrie to a location that is far enough away from Peggy, Carrie's proxy Focus will not reach Peggy's location in the GPS space, even if the configuration of Carrie's Focus with Peggy's projected presence suggests that it should.

It is easier to visualise how this particular configuration of projection points and users might occur when considering the case of reasoning across two Continuous Symmetric spaces as illustrated in [Figure 23232216].

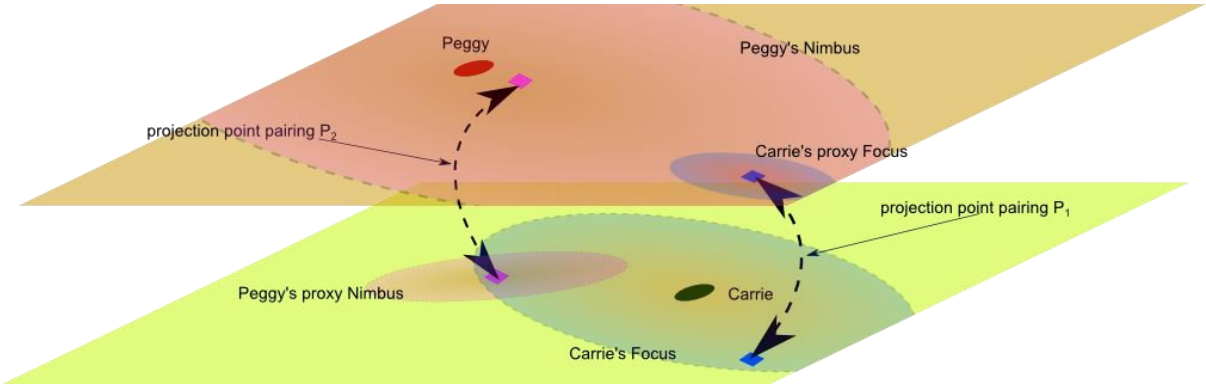

Figure 23 – Projection Point relationships between two coninuous spaces

Since our model dictates that projection point pairings signify two points in two different spaces which are to be considered the same point in some

aggregate space, it seems that this inability for Carrie's aura to flow through the projection point which Peggy is using to project her presence creates a flaw in the awareness relationship between the users. In her native space, Carrie's aura reaches one side of the projection point whose pair is very close to Peggy's location, implying that it should cover Peggy's location in the aggregate space, but it can never cover Peggy's location in this configuration due to the existence of a projection pairing closer to Carrie and the preferential selection of proximate projection points. Carrie's presence cannot travel through the projection point that Peggy uses and into the GPS space at that point extremely close to Peggy.

Whilst it would certainly be useful to investigate a version of our model that allows presence to project into a single space through every projection point pairing that a user's aura covers, endowing the user with multiple projected presences in a single space, and aggregates the multitude of awarenesses gathered from these presences into a single awareness level and type, we believe that exploration to be outside the scope of this thesis whose initial aims are to investigate the utility of a very simple model. Further, we would argue that since we are using projection points to construct an aggregate space, a configuration of projection points that causes the situation described above to arise can be said to be either reflecting intended properties of the aggregate space, or an unavoidable consequence of the scarcity of data relating to the relationship between the two spaces. Semantically, a projection point/user presence configuration of the form in the example, stretches the distance in the aggregate space from Carrie to Peggy via Peggy's projection point to a distance of infinity, whilst preserving the distance from Carrie to Peggy's projected location as defined in the distance metric. In this way, projection points may imbue aggregate spaces with a form of structure similar to that of asymmetric discrete spaces, if that is what a developer desires. And although these configurations and structures can arise when not desired, they do so as a consequence of our model's support for context representations and relationships between those representations which must be discovered or allowed to evolve at runtime. We assume looking at [Figure 23232216] that Carrie's projected focus should be able to reach Peggy's location because the

image of the two spaces aligned with each other gives us what appears to be perfect knowledge of the relationship between them. However, the reason Carrie's projected Focus does not reach Peggy's location in the model is because the only information stored in the model about the relationship between the two spaces is that encoded in the two pairs of projection points and we do not actually know how the spaces relate outside of those two projection point pairings. The experiences of developers investigating seams in Pervasive computing certainly give the impression that relationships between spaces are not always so neat as the perfect alignment depicted in Figure 22222115 would suggest. Our assumption if this configuration arose during program execution would be that the information encoded in those two pairs of projection points is the only information currently known about the relationship between the two spaces and that it represents the most accurate depiction available of the relationship between the spaces.

Where these sorts of confounding inter-space presence relationships exist, however, we note that introducing more projection point pairings, further information about how the two spaces relate to each other, to the model can ameliorate the problems raised by the class of projection point/user presence configurations illustrated in [Figure 23232216]. As our model is designed to support applications which use context representations whose relationships are discovered at runtime, we can assume that over time during program execution, more data about the relationship between the two spaces will be gathered and encoded into the model, and consequently the occurrence of projection point configurations which produce counter-intuitive awareness results will decrease.

A worked example

A Context-aware sight-seeing application provides tourist information for every building and site of interest in a city. The application uses GPS to position these boards in a Continuous Symmetric model space, other location technologies are incorporated into the application using our model. The Nimbus of each board represents the area within which the application

developers feel information to tourists about the site would be most useful and relevant. The Focus of each user represents the area which the user is interested in seeing information about. The application developers set the distance metric of the GPS space to a function that calculates the Euclidean distance between points, the function O(A, P) which scales aura being projected out of the space to a neutral value and the function I(A,P) which scales aura being projected into the space are set as:

$$O(A,P) = A \times 1000$$

$$\mathrm{I(A,P)} = \frac{\mathrm{A}}{1000}$$

Carrie uses the application with a mobile device which depends on Wi-Fi location technology. The application builds a representation of the distribution of Wi-Fi networks across the city and this is modelled as a Discrete Symmetric space. The space uses a distance metric which depends on calculating the total value of weighted edges between two vertices. The functions O(A,P) and I(A,P) for the space are set at:

$$\mathrm{O(A,P)} = \mathrm{A} \times 5$$

$$\mathrm{I(A,P)} = \frac{A}{5}$$

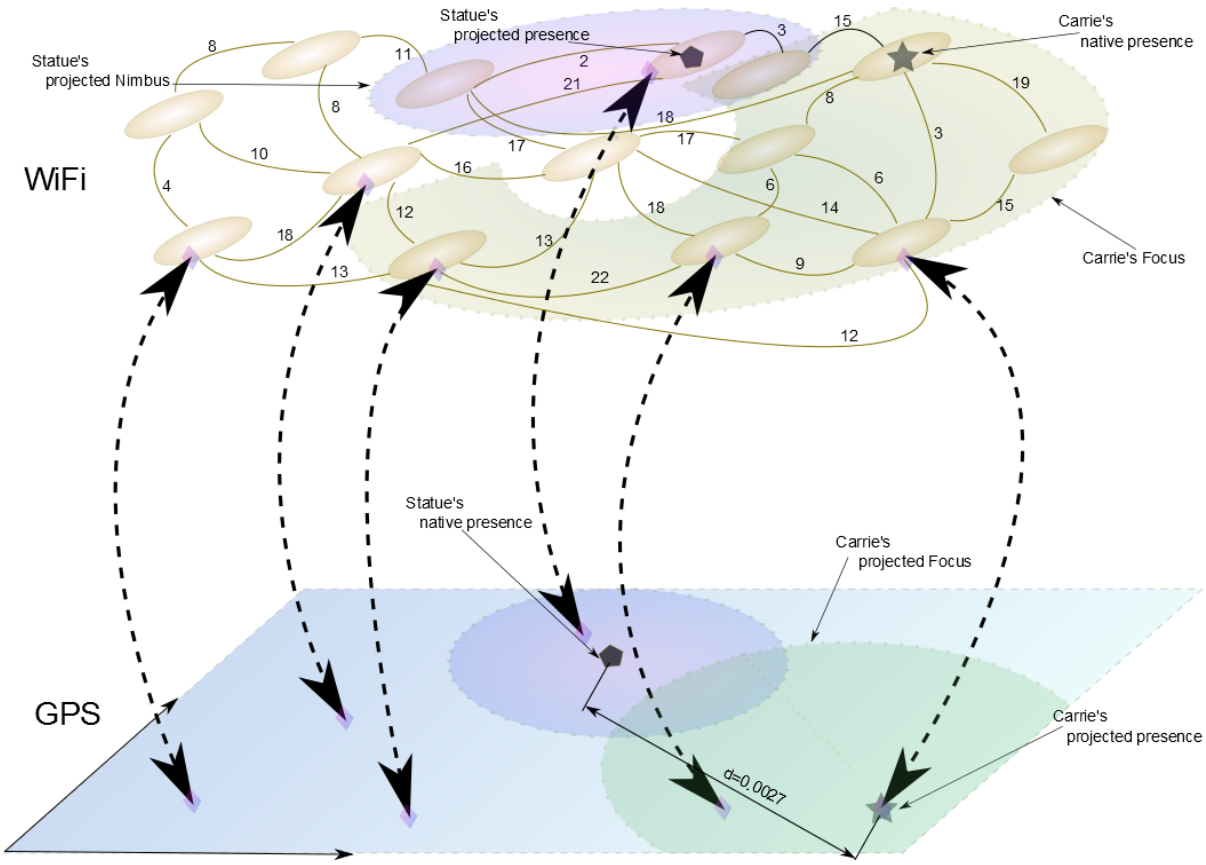

Figure 24 – Through their projected presences, Carrie gains awareness of Peggy in both the GPS and Wi-Fi spaces.

Imagine that so far during the application's deployment, several projection point pairings have been detected and added to the model. [Figure 24242317] shows a subsection of the Wi-Fi space, the GPS application space and the projection point pairings between them. The relationship that is being constructed between the two spaces is one of overlay, the application developers ideally wish to construct an aggregate space in which the two spaces are aligned exactly with each other.

Carrie's device locates her at vertex W_1 in the Wi-Fi space. Her aura's size is set at a value of 16 and it can be seen in [Figure 24242317] that this covers several projection point pairings. The closest of these is the pairing located at W_2 of distance 3 from Carrie's location. The model projects Carrie's presence through the pairing into the GPS space giving her a proxy presence located at (1.01, 0.0012) as illustrated at the bottom right of [Figure 24242317]. Carrie's aura is reduced by the distance between her location and the projection point $D_C = 4$ to a value of 12 and transformed as it is projected out of the space to a

neutral value of 2.4, it is transformed a second time as it is projected into the GPS space to F'=0.0024.

The sight-seeing application has attached some information to a statue located at (1.0074, 0.0022). As the statue is rather small and cannot physically be seen from far away, it is given a Nimbus of 0.0008. [Figure 24242317] shows that this Nimbus covers a single projection point at (1.0073, 0.0022) which is a distance of 0.0001 from the statue. The statue's presence is projected through this pairing into the Wi-Fi space to the location W_3. The statue's Nimbus is reduced by the distance between the statue and the projection point D_p to a value of 0.0007, it is then transformed as it is projected out of the GPS space to the value 0.7, and transformed as it is projected into the Wi-Fi space to the value N'=3.5. [Figure 24242317] shows both the presence and proxy presence of Carrie and the statue in the Wi-Fi and GPS spaces.

The GPS distance metric gives the distance of Carrie's proxy presence from the statue's native location as $D_G = 0.0027$. Applying the awareness type evaluation, the model determines that the awareness existing between Carrie's proxy and the statue is low. Applying the awareness level equations for awareness between a user and a user proxy to their presence configuration gives the values

$$A_{C_1} = \frac{F' + N - D_G}{N} = \frac{0.0024 + 0.0008 - 0.0027}{0.0008} = 0.625$$

$$A_{P_1} = \frac{F' + N - D_G}{F'} \times \frac{F - D_C}{F} = \frac{0.0024 + 0.0008 - 0.0027}{0.0024} \times \frac{12}{16} = 0.15625$$

The Wi-Fi space's distance metric gives the distance of the statue's proxy presence from Carrie's native presence as $D_W = 18$. Applying the awareness type evaluation again, the model determines the awareness existing between Carrie and the statue's proxy is low. Applying the appropriate awareness level equations gives the values

$$A_{C_2} = \frac{F + N' - D_W}{N'} \times \frac{N - D_P}{N} = \frac{16 + 3.5 - 18}{3.5} \times \frac{0.0008 - 0.0001}{0.0008}$$
$$= 0.375$$

$$A_{P_2} = \frac{F + N' - D_W}{F} = \frac{16 + 3.5 - 18}{3.5} = 0.428$$

The model combines the two awareness types to give an overall awareness type between Carrie and the statue of Low. Since A_{C_1} is the larger of the projections of Carrie's Focus into the statue's Nimbus and A_{P_2} is the larger of the projections of the statue's Nimbus into Carrie's Focus, they are selected for aggregation into her total awareness level of the statue which is:

$$A_T = \frac{A_{C_1} + A_{P_2}}{2} = \frac{0.625 + 0.428}{2} = 0.5265$$

This awareness level and type is used by the application to determine the type and nature of the information about the statue that should be delivered to Carrie. When the awareness between a user and an object or site of interest is low, the application provides the user with a photograph of the object so that the user might recognise it as they approach it, the photo is selected based on the awareness level, with a photo taken from a further distance, showing more of the surrounding area, selected for lower awareness values, versus a more close-up photo for high awareness levels. Also delivered are a map of the object's location so that the user can more easily find the object, and a one-line blurb about it so that they can decide if it is of enough interest to them to seek out. The awareness type and level are delivered to Carrie with this information payload so that her client application can filter out information about objects or sites she is not interested in. For example, Carrie may set her preferences to only display notifications about sites or objects which are within her area of interest (her Focus), i.e. those which she has Active awareness of.

Conclusion

We have argued that it is useful to represent context spatially, and to use spatial reasoning to reason about context. Specifically, we chose to apply Benford et. al's awareness approach to create a model that performs this spatial reasoning

about context. In the previous chapter we described the types of space that arise from a spatial treatment of context, and in this chapter we further explored their properties and how those properties dictate the way in which we represent aura and presence in our model. We examined the importance of continuity of context and thus awareness in our model, and presented the details of our mathematical approach to providing this continuity to users through the calculation of Awareness type and level within the various spaces. We also explored the challenges that arise when calculating awareness level and type within Asymmetric Discrete spaces and our approach to tackling them.

We had proposed the projection of user presence into other spaces as a method for enabling awareness across spaces and thus reasoning about context across contextual representations and in this chapter's section on projecting user presence we explored the use of projection point pairings—pairs of points, or sets of points, drawn from two spaces which represent the same point, or set of points, in some aggregate space—as a method for structuring the relationships between spaces and enabling users of one space to gain presence in another. We explored the spatial and contextual implications of various approaches to structuring projection points and projected presence in order to illuminate the reasoning behind our chosen approach to constructing projected user presences.

The creation of an additional presence in a second space can give a user two sets of awareness types and levels for another user, one in their native space and one in the space which they are projected into. These sets much be aggregated in order to determine the awareness that a user has of an entity in another space. In chapter 3 we laid out the methodology for aggregating awareness types across spaces, and in the final section of this chapter, we detailed our model's mathematical approach to aggregating the multiple levels of awareness that arise when we reason about the relationship between two users across two spaces. Having presented the core of our mathematical approach to aggregating awareness levels, as with the calculation of awareness type, we then discussed the challenges presented by various edge cases for awareness level calculation and how we tackle them.

Finally, we provided a selection of examples that illustrate how our mathematical model can be applied to various scenarios in which developers of context-aware applications might wish to create structured relationships between different context elements and their representations.

Having laid out our theoretical and mathematical models, we present in the next chapter an implementation of these models in the form of a context broker system and a set of APIs which allow developers to take advantage of the context broker's awareness reasoning services.

Glossary of New Terms

Continuity of Context is when small changes to a user's presence do not result in large changes to her Awareness.

Chapter 5

The SPACES Platform

Introduction

In the previous chapter we presented the mathematical framework that underpins our SPACES model for reasoning about context. We discussed how, given two representations of context, which may be spatial or non-spatial in nature, we can reason about the user relationships that exist across them by modelling them as spaces and creating links between the spatial model of each representation, which we call projection point pairings, and projecting each user's presence through these links into a second representation.

Our model is designed to support developers who create context-aware applications which employ multiple elements or representations of context. Although simply having a framework within which to consider our treatment of context is helpful, of more practical use is a platform upon which developers can construct context-aware applications that use the structures of our model and, adjunct to that framework, a system which removes from developers the burden of reasoning about context. In order to demonstrate our model, we constructed such a system—a context-broker, built atop a tuplespace, that determines the type and value of awareness between two users from information inserted into the tuplespace by developers. To organise and encode this information as our model does, we created a set of APIs which developers must use to access the context-broker service. In this chapter we describe the structure of this Spatial presence & Awareness Evaluation Service (SPACES), how it supports developers who need to reason within and across different representations of context, and its relationship to the various aspects of our model. We also detail our algorithms for reasoning about presence and awareness—these include implementation of our model's central awareness type and level calculations, presence projection calculations.

For example, our model divides users into Producers and Consumers. Producers are those who broadcast information, provide access to services, or execute behaviours, for a user who has awareness of them. Consumers are those users who gain awareness of Producers and receive their information, access their services or benefit from the behaviours they execute. Given a particular awareness type and level, our model specifies that a producer may adjust the form or value of the payloads it delivers to a Consumer, and that a Consumer may filter or manipulate those payloads that it receives. We describe a selection of examples that show how developers might implement this procedure.

Our model's ability to reason across context representations is dependent on the discovery of projection point pairings –pairings of points from two separate spaces which represent the same point in some aggregate space. The detection and creation of these allows us to build a point-based description of the relationship between two spaces. Since detection of these relationships can be dependent on the context-representations used or even the application, our model and our context-broker leave the majority of approaches to this detection to developers. However, we do provide a basic procedure, proceeding from our discussion of the nature and meaning of projection point pairings in Chapter 3, and the scenarios in which they might be employed, in the Context-Broker and we detail its mechanism and the way in which it lends itself to supporting a more general collection of information about the relationships between the two representations.

The Architecture of the System in Overview

Our model's aims are to provide a spatial system for reasoning about context to support application developers who are creating context-aware applications that use one or more elements and representations of context, where those elements or their relationship to each other may be unknown or partially known to the developer during development and runtime. In addition to their frequent dependence on evolving data sets, context-aware applications are often mobile. Games like Savannah, Can You See Me Now and Google's Ingress depend on users who are able to move around physical gamespaces which stretch

respectively over playing fields, cities and entire countries. Augmented reality applications such as Cityview; and social, context-aware applications such as Highlight are specifically conceived to support users during trips and excursions. All of these applications are structured in client-server architectures to one extent or another, but the increasing level of resources delivered by mobile technology platforms provides the opportunity for approaches to application architecture that are more distributed. Co-operative games, for example, may perform larger amounts of data processing on the client device as opposed to the server, and those that are co-located may pass some messages over local, ad hoc Wi-Fi networks rather than via a server over slower 3G connections, in order to preserve the responsiveness of the game for users.

The communication and collaboration required between applications in order to access models of context representations and reason across them raises the question of just where our model should be implemented. Implementing the model exclusively as a platform on top of which individual applications are built, so that context-reasoning is completely distributed across the applications using the platform, introduces the challenge of maintaining projected users' presences and performing cross-representation context reasoning through the collaboration of what may be multiple mobile clients.

Alongside this is the issue of collaboration in representation maintenance. Collecting information to build new representations of context elements, or maintain evolving representations, can require significant investment of resources, which developers may not have. One way to achieve this is to crowdsource those resources, but convincing users to part with, for example, their time or extended battery life on their mobile device in order to sample and record information about a new context representation is difficult.

Because it is undesirable for developers to have to create separate data-gathering games or applications to go with any context-aware application they conceive, it is reasonable to assume that, if they do not otherwise have that information to hand, e.g. when using GPS the representation's underlying data set is well understood; there will be instances where a developer receives information about the structure of context-representations from third party

applications. As discussed in the previous section, classes of applications can require domain-specific models of context-representations. Converting representations into these models can in itself be a non-trivial task and, in the case of representations that change or evolve over time, requires ongoing resource investment in order to maintain the model. A single developer performing this conversion specifically for his application does not benefit the ecology of context-aware apps in the same domain that might make use of such a model. Unless the developer makes the structures he creates available to others, those developers will have to perform the same such analysis before they can apply context-reasoning to the representation. This structured information can be shared between applications on an adhoc basis, but where representations are subject to change, applications then incur additional resource overhead ensuring that everyone has the most up to date representation model and negotiating amongst each other as to who is responsible for its accurate upkeep. The onCue toolbar described in [DIX, CATARCI et al.'06] offered a way to automate a similar context-aware process in a desktop environment by bringing together a set of text-recognisers and text-processing services to examine clipboard content for its context e.g. Recognising a Telephone number, and offer services related to that context.

Implementing the context-reasoning procedures of our model as a service provided by a centralised Context-Broker application can not only remove from developers the burden of reasoning about user context, it can provide a way to share third party information about Context-Representations in the spatial structures that are meaningful and appropriate for reasoning about context, without requiring applications to negotiate the exchange and maintenance of those structures amongst themselves. For example, one developer may construct and maintain a model of an NFC location system for a city using a game similar to Google's Ingress, whilst another developer uses the system to provide user state for his context-aware application. The second developer does not need to know anything about the structure of the representation or its space in order to make use of it because the first developer maintains those structures and submits information about them to the Context-Broker, which can be tasked to perform all the required calculations upon

them. Further, the developer who constructs the NFC location system for his city may collaborate with another who constructs a similarly structured system for a neighbouring city so that the two maintain together a single representation of NFC location across the two towns. In this case, an application developer using the NFC location system need not even be aware that the spatial model is being constructed and maintained by two separate applications. This context-reasoning-as-service approach also supports a range of application architectures from server-based to mobile and distributed.

There are myriad context elements that applications might require our Context-Broker to reason about: location, temperature, and social connection, to name but a few-- and each context element may have multiple representations, for example location as GPS, GSM and Wi-Fi; and social connection through various social networks such as Facebook, Twitter, and Tumblr. Additionally, developers can model representations in a variety of ways, dependent on their applications' specific function, what we term as the application's domain. GPS, for example, can be modelled as a Continuous Symmetric space for use with an application like Highlight, but for a hiking app which estimates a hike's length based on the difficulty of navigating the terrain, the same representation would be encoded as a Continuous and Asymmetric space, since hiking a path uphill is much more difficult that hiking it in the opposite direction. This specificity of context to the domains of individual applications, or classes of application, makes it unrealistic for our Context-Broker to itself provide a service that attempts to implement the construction and maintenance of the entirety of these spatial models. Instead, we split responsibility for this construction and maintenance of model spaces between the Context-Broker and the applications that will use its services.

Similarly, the Context-Broker does not itself sample user states and encode them to the form of points in a space's Point Set. This is also a domain-specific task and is thus left to the user's client, to an application server, or to a third party. Allowing applications to encode user state themselves, provides them with the leeway to perform corrections to, and manipulations of, the user state. This gives developers an opportunity to manage the impact of sensor seams on

the user's experience, for example by smoothing jitter or drift in a GPS reading, or by returning from the distance metric the smallest possible value of a distance between two points where there is uncertainty as to the definitive distance—what Benford might refer to as an optimistic approach to uncertainty. Once again these encoded user states are structured and supplied to the context-broker using the API. A CityView user's state would, for example, be encoded as a Consumer object with a Continuous location and a Focus Aura. An augmented reality cinema billboard in the CityView application could be encoded as a Producer with a Continuous location, a Nimbus Aura and a Payload which connects mobile clients to a service that displays film posters, reviews, and screening times. These structures map to the user point of presence Location, Aura, Intent tuples in the model, with the consumer intent implied in the Dataspace tuple by the presence of a Focus object.

Our model abstracts context, and individual user states within a specific context, to sets of features in order to allow us to reason about them in a systematic manner. The process of reasoning about these features: determining user awareness based on mathematical calculations about the spatial relationship between two user presences, is described by the model acting on these abstract, external features of its components. It is this responsibility to reason and act on the external features of a representation's spatial structures, and to reason about the interplay between those structures, that we delegate to our Context-Broker. Since it is application developers who are most familiar with the workings of their context-representations and the ways in which those representations map to the types of spaces in the model, we require applications to make their representations' values and processes available to our Context Broker by encapsulating them in these implementations of spatial structures whose features are set by our model and the API.

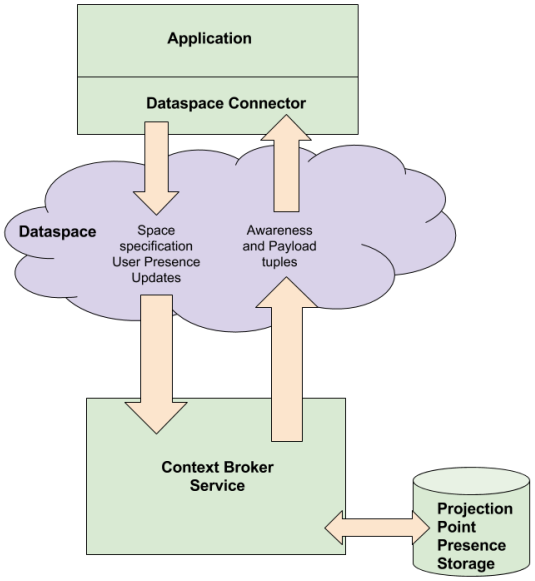

Figure 25 - The high-level architecture of the SPACES system

The model also specifies how to make use of the results of awareness reasoning, by conceptualising context-aware applications as populated with entities that can take the role of Producers, who distribute customised payloads to those who have awareness of them in the model, or Consumers who receive payloads and may, based on their awareness of the Producer, manipulate them further. Awareness type and level between two users provide cues as the manner and degree of manipulation each user should apply to the payloads passing between them, but developers implementing those payload objects are free to choose how exactly they are manipulated for a given Qualitative or Quantitative Awareness. The high-level architecture of the system can be seen in Figure 25252418 which depicts an application implementing the API and using the DataspaceConnector communicating with the Context-Broker. The Dataspace supports receiving updates from multiple such applications for users in the same spaces.

It is our developer's API that creates the framework which applications use to encapsulate these details of their context-representations' behaviour. The API provides a set of interfaces that describe the external features of the model's spatial structures and applications must provide implementations to the Context-Broker to take advantage of its context reasoning service.

Aspects of the Model Realised in the API

The cornerstone of the model is the concept of context-representations as Spaces. Spaces are described by their Distance Metric function; their Point Set; their individual Points, their Bounding Box, which is used to describe contiguous sets of points; and by their Aura manipulation functions that are involved in adjusting the Focus or Nimbus of users' proxies projected out of, or into the space. Our API which is written in Java, presents these structures as interfaces to be implemented or abstract classes to be extended, what follows is a description of the structures of the model and their features exposed to applications via the API.

The Space

Space in our model is the structure wraps all the elements that describe the structure of a context representation's model. In our API a Space is both a wrapper that bundles these objects into a meaningful unit, and a structure that provides the spatial services required by the model, specifically those which may employ multiple of the space's structures to deliver their result. For example, the BoundingBox structure models groups of points in the space that participate in projection point pairings and since the method that returns the centre of a Bounding Box may reasonably require use of the Distance Metric and the Point Set, the space types provide references to these to the BoundingBox. Similarly, the Distance methods of the Metric, the membership methods of the Point Set, and the manipulation methods of the Aura manipulation functions are encapsulated by methods that the space provides. The abstract class Space has four abstract subtypes and a Space's sub-type is determined by the subtypes of its Distance Metric-- Symmetric or Asymmetric-- and its Point Set-- Continuous or Discrete, thus a DiscreteAsymmetric space type has amongst its members a Discrete Point Set and an Asymmetric Distance Metric.

The Space contains members of the classes: BoundingBox, DistanceMetric, PointSet, Point, IFunction and OFunction. It offers the following methods:

BoundingBox newBoundingBox(Point p); which given a point will return a new BoundingBox object containing just that point

BoundingBox addToBox(Point p, BoundingBox b); which given a BoundingBox and a Point will add that Point to the BoundingBox object and return the updated BoundingBox.

Dist getDistance(Member m, Point p); or **DoubleDist getDistance (Member m, Point p);** which act as the distance functions for a Symmetric and Asymmetric space respectively, where **DoubleDist** is an object which wraps two distances – one from the Member to the Point and the other from the Point to the Member. The model supports the calculation of distance in two cases: between two users whose locations are points, and also between a user and a projection point in order to determine if her Aura intersects the projection point. Since a projection point can be either a Point or a Bounding Box, we use a superclass Member of which the classes Point and BoundingBox are subclasses, therefore **the getDistance** method takes a Point and a Member which can be either a second Point or a BoundingBox.

Boolean isMember(Point p); which given a Point will return true if the Point is a member of the space's point set

Boolean isStructured() is provided by DiscreteAsymmetricSpace objects and returns true if the space has a structure that may affect its Awareness calculations.

DoubleDist getDepth(Cpresence c, Ppresence p); is a method provided by the DiscreteAsymmetricSpace for discovering the projection of the Focus into the Nimbus and vice versa in a structured DiscreteAsymmetric Space.

The set of methods: **Focus focusOut(Focus f, Point p); Nimbus nimbusOut(Nimbus n, Point p); Focus focusIn(Focus f, Point p);** and **Nimbus nimbusIn(Nimbus n, Point p);** give access to the IFunction and OFunction for manipulation of projected Aura.

Finally, the space allows updating its structures via the methods **setPointset(Pointset p); setMetric(DistanceMetric d);**

`setIFunc(IFunction i);` `setOFunc(OFunction o)` and `setBoundingBox(BoundingBox b);`

Figure 26262519 shows the space class and its members alongside their subclasses.

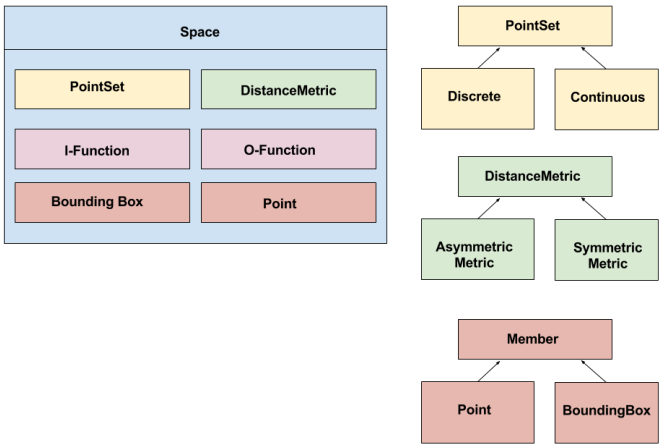

Figure 26 - The Space Class and its members

The Distance Metric

A space's Distance Metric can be either Symmetric or Asymmetric. A Symmetric distance metric provides a method, called by the space when the Context-Broker calls its `getDistance` method, which given either two Points, or a Point and a Bounding Box, returns a single scalar value representing the distance between the two, to support the metric in calculating the distance the space also passes it a reference to its current PointSet object. An Asymmetric distance metric provides a similar method which takes either a pair of Points or a Point and a Bounding Box with a reference to its PointSet and returns a pair of distances, the first distance, signifying the distance from the first argument to the second and the second distance signifying the distance from the second argument to the first. The Distance Metric structures are provided as abstract classes and developers are expected to extend one or the other as part of their space's description.

The PointSet

The point set, referred to in our API as PointSet can be either Continuous or Discrete. Our model employs different approaches to the calculation of user awareness dependent on the type of space the calculation is being performed within, and it is the PointSet's type, together with the type of the Distance Metric, that provides the Context-Broker with cues as to which of these approaches it should employ. The Point Set also provides a method **`isMember(Point p);`** that given a Point object will return a boolean indicating if the point lies inside the space-- useful to the Context-Broker for verifying that third-party context-sensing tools are reporting user states that are valid in the application's chosen context-representation. Discrete Point Sets provide the Context-Broker with three further methods for use where the Space is also Asymmetric these are isStructured, isWeighted and getEdgeSet.

The **`isWeighted();`** method returns true if edges in the point set's structure are weighted. In this case the Space expects the Edges to be triples of the form (V_1, V_2, W) indicating that a directed edge exists from vertex V_1 to vertex V_2 of weight W.

`Edge[] getEdgeSet()` method returns an array of either ordered pairs or ordered triples describing the edges in the space. This is used by the Space to determine the depth of projection of Focus into Nimbus and vice versa when satisfying a call to its getDepth() method using the greedy breadth-first algorithm described in Chapter 4. If the PointSet is not structured—i.e. **`isStructured()`** returns false—**`getEdgeSet();`** may return a null value since without an Edges structure on the pointset, the Context-Broker should employ the regular method of calculating awareness which uses on the distance metric. The Edge object is simply a wrapper with the members Point1 Point2 and Weight.

Dependent on the type of space he is working with, a developer can choose to extend the abstract Continuous PointSet class, or the abstract Discrete PointSet class as part of his space's description.

The Point

If a user's context describes her state, then a context-element's representation is the set of all possible states that a context-sensing or gathering technology could assign to a user. The Point in our model provides an abstraction for each individual state that makes up this set. It is employed as input to the Distance Metric, as Location for Consumers and Producers, and as members of projection point pairings or of Bounding Boxes. Since a point can be anything from a simple number to a set of strings, to a complex object, as a type in our API it acts as a wrapper to allow the Context-Broker to retrieve it from one method and pass it to another, without having to handle these differences explicitly. Particularly, the Point interface allows the Context-Broker to handle both Continuous and Discrete Points using the same code. The abstract classes representing Continuous Points and Discrete Points implement the Point interface, but defer implementation of its methods to their application-specific descendants.

As the Context-Broker must be able to identify the space that each Point belongs to, the Point interface provides methods to set and retrieve the uniquely identifying name of the Space the Point belongs to. In addition to this, part of the Context-Broker's approach to detecting Projection Point Pairings requires maintaining a list of previously reported locations for a user with the time they were reported, and our implementation does this by storing those locations in a local database, therefore the Point interface also provides a method that allows the Context-Broker to retrieve an encoding of the Point's value as a text string. Although the Point included as a member of the Space object is not specifically employed in any calculations, its existence there ensures that its class definition is present and loaded in the Context-Broker before any user presence objects containing Points of that class are received.

The Bounding Box

The Bounding Box provides a way for the model to group contiguous Points of a Point Set for the purposes of linking them to other spaces using Projection Point Pairings. As we allow developers to implement the BoundingBox object themselves, this may be a true bounding box, a convex hull, or simply a

collection of points depending on the developer's preference, so long as the methods of the interface are satisfied. Imagine the Context-Broker discovers that the point (0,0) in a GPS space has a relationship to the cell 98654 in a GSM space, this relationship is recorded in the Context-Broker's model as a projection pairing of the two individual points. When the Context-Broker later discovers that the point (0,0.001) also has a relationship to cell 98654, that point should be added to the existing pairing. This is done by wrapping the two GPS points in a Bounding Box which presents them and all the points in between them as a single structure. Projection point pairings are called into play by the model when determining if a user proxy should be created and during the creation of a user proxy to determine its location and Aura size.

The BoundingBox class provides four methods, the first two of which are:

`getNewBox(Point p, DistanceMetric m, PointSet ps);` which the Space uses to create a new BoundingBox

`addToBox(Point p, DistanceMetric m, PointSet ps);` which the Space uses to add a new point to the Box it is called on.

The space passes references to the PointSet and DistanceMetric in these methods so that if the BoundingBox class requires them in the creation or updating processes they are available to it. To create a new BoundingBox from two GPS points, for example, the ContextBroker first calls the ContinuousSymmetric space's **`newBoundingBox(…)`** method with the first Point which in turn calls its BoundingBox member's **`getNewBox(…)`** method with Point and the Pointset and Distance Metric references. The space returns the results of that method call (the new BoundingBox) to the ContextBroker which then calls the Space's **`addToBox(…)`** method with the newly created BoundingBox and the second point object as arguments. To satisfy the call, the Space returns the results of calling the Box's **`addToBox(…)`** method. Further points can be added to the Box by repeatedly calling the Space's **`addToBox(…)`** method for each new Point. Note that the ContextBroker cannot call the **`addToBox(…)`** method itself as it does not have access to the Space's PointSet and DistanceMetric objects to pass the method.

The third method `getPoints();` returns an array of the Points that comprise the box, which is every point that has been added to the box through the first two methods. This is used in the process of managing Projection Point relationships in the case that two BoundingBoxes must be merged.

Bounding Boxes are used by the Context-Broker both to maintain the record of relationships between spaces and in the creation of proxy presences. When determining if a proxy should be created in a space, the model requires that the Context-Broker find the set of projection points linking to that space whose endpoints in the user's space lie within her Aura, and from this set determine the closest point to the user's location. Where a projection point's endpoint in the user's space is a Bounding Box, this involves requesting the distance between the Bounding Box and the user's location, however, the Context-Broker does not need to access any internal feature of the Bounding Box to do this. The Distance Metric may require access to representation-specific features of the Bounding Box in order to perform its calculation, but these are specific to the developer's implementation of the calculation of distance in the model, and therefore can be considered internal features of the space's structures, which are not exposed to the Context-Broker. A developer has the option to implement this `getDistance(…)` method of the DistanceMetric as distance to the boundary of the box, distance to its centre or some other measure, dependent on the sorts of operations the DistanceMetric can support.

On the other hand, when determining a proxy's location in the space which it is projected into, which we call its target space, the model dictates that where the projection point pairing's endpoint in the target space is a Bounding Box, the proxy's location be taken as the point at the centre of that Box. To do this, our Context-Broker requires access to a method which can return the centre of a Bounding Box. This is accessed via the BoundingBox's fourth method: `Point getCentre();` which developers must implement. Since Bounding Box objects will be often handled by the same code as Point objects by the Context-Broker, the Bounding Box extends the same Member class as the Point.

The Aura Manipulation Functions

Our model specifies that in order to reconcile the differences in scale between spaces, when a user's presence is projected out of its native space via a Projection Point Pairing, its Aura must be manipulated into a neutral scale (which can be agreed upon between developers) and when it enters the projection pairing's target space this neutrally scaled aura must be manipulated to match the scale of the space it enters. We call the function which manipulates Auras leaving a space, the space's O-Function, and that which manipulates Auras entering a space, the space's I-Function. Because our model allows for scale to vary within a space, such as in GSM spaces where a single broadcast cell can cover distances from a few metres to tens of miles, these functions are dependent not only upon the size of Aura but the location at which the transformation takes place. In our API each of the abstract Aura Manipulation classes provides an Aura adjustment method which is called by the Space when executing its `focusOut(…); focusIn(…); nimbusOut(…);` and `nimbusIn(…);` methods. This method `scaleAura(Aura, Member);` takes as its arguments the Aura object and the Bounding Box or Point endpoint of the Projection Point Pairing through which the user presence is projecting out of, or into, the space. It returns the modified Aura which is then returned to the Context-Broker by the Space.

Users

Users and the objects defining their presence are the other major component of our model. Their state, encoded in the model as their intent, Aura and location within a space, informs all of the Context-Broker's awareness reasoning. The model splits users by their intent into Consumers, those who receive payloads based on their awareness; and Producers, those who create and distribute payloads to Consumers with awareness of them.

A user's location must be a Point object from the space that the user is attached to. Aura which wraps a single scalar value, provides methods for setting and accessing its value so that the Context-Broker can use it in its calculations, and create new Aura for projected presences. Aura is implemented by the API as two final sub-types: Focus and Nimbus, whose typing supports the model

restriction that Awareness occur only between a Consumer and a Producer, rather than two Producers, or two Consumers.

The model states that where a Consumer has a particular type and value of awareness of a Producer, that Producer must provide the Consumer with a version of their payload that has been customised according to the type and value of the Consumer's awareness. In the API, Producer Payloads are both encapsulations of information, services, and/or behaviours, and factories which produce versions of themselves customised according to the awareness details supplied to them. To this end, they provide two methods, one which a Consumer's client application can call to retrieve the value of the payload itself, and one which the Context-Broker can supply with Awareness details in order to retrieve a customised version of the wrapped Payload object for delivery to a Consumer. As the Context-Broker manages the presence of users in the model, and those presences, including their projections, span multiple spaces the API does not provide wrapper classes for users. The classes of Consumer and Producer are internal to the Context-Broker and populated by both the information the Broker receives from applications and the information it deduces from those states and the relationships between spaces.

Employing the implemented Spatial and User classes in the Model

The model structures of the API that describe Spaces and Payloads are provided either as interfaces which Developers can implement or abstract types that they must extend and it is in the implementation and extension of these structures that developers describe the operational details of their application's spatial and payload structures. The Context-Broker, like the API is written in Java, and since these application-specific implementations are introduced to it at runtime, it employs classloading to access them. When these objects are delivered to the Context-Broker, they are always accompanied by a URL through which their Class definitions can be accessed by the Broker's Classloading modules.

Projection Points

Our model uses Projection Point Pairings to encode information about the relationships between spaces. The Pairings are pairs of points, or pairs of sets of points from two different spaces which represent the same point in some aggregate space. For example a pairing could encode that a particular set of GPS points cover a GSM node's broadcast range, that a twitter username is used by the same person attached to particular Facebook identity, or even trivially that two temperatures given in Fahrenheit and Celsius are the same. In our model, we can think of these as spatial relationships wherein two different representations of the same underlying data source---in the first instance location, in the second identity and in the third temperature-- are lain directly over each other. The aggregate space that we refer to when we talk about Projection Points is this underlying data source. However, an aggregate space can take other forms. Some of these include the mapping of one physical space over another, the mapping of an imaginary virtual space over a physical space, and the stitching together of two spaces at their borders.

Imagine an augmented reality application to broaden museum access allows the mapping of one of the museum's rooms to school's auditorium so that visitors to the school, viewing the auditorium through their augmented reality mobile device can both explore the museum's exhibits and interact with those current visitors to the museum who are also using augmented reality devices. This mapping lays the physical space of each museum room over the physical space of the school's auditorium. In our model each space would maintain its own location representation and the model would create the relationships required between these in order to allow users at the school to access the exhibits and visitors located at the museum. If two museums were to collaborate in this way to allow visitors to one exhibit to observe another exhibit this might involve stitching together of two representations of physical spaces at their edges. And a museum might even go so far as to create completely virtual exhibits whose same layout is projected over auditoriums in multiple venues across the world, mapping the imaginary space that contains the virtual exhibit over multiple physical spaces. In these examples, developers are aware of how the spaces

involved match up to each other and Projection Point Pairings can be simply specified. In other applications however, these relationships are not constructed arbitrarily by developers nor are they readily apparent, instead they must be discovered.

Projection Point discovery

Many context-aware applications have used War-Driving to perform the discovery of relationships between location representations, and we have already mentioned application like Google's Ingress that gamify this process. Although a full investigation into the different ways in which these relationships can be discovered for various types of context is outside this work's scope, our context-broker does provide one method of discovery. Given two different context-representations whose relationship to each other we wish to map, approaches such as Google Ingress and War Driving depend on users being in possession of the technologies that allow them to access both representations, for example in mapping GPS and Wi-Fi, we depend on the user's device having both GPS and a Wi-Fi transmitter. Readings are then taken in each representation simultaneously and because the application knows that those readings were taken at the same time -- co-temporally—and on the same device, it also knows that they must be taken in the same physical place-- co-locationally. Our Context-Broker detects the existence of relationships between spaces by checking for these co-temporal readings in the Consumer and Producer location updates it receives and making the assumption that where they exist for a single user, they must be from the same device, or from different devices that are carried by that user on their person and therefore co-located. Since applications can provide information about multiple spaces to the Context-Broker, a single application might be the source of the user's presence updates in two different representation spaces. It might also be that the user is running two different applications which use different representation technologies and employ the same Context-Broker service to reason about their representation space. In this initial implementation of our model, we assume all applications with access to the dataspace are trusted as non-malicious.

To manually construct a relationship between two spaces, a developer then can create a dummy Consumer and submit the appropriate point relationships to the Context-Broker as pairs of Consumer location updates, whose spaces are those the developer wishes to link and which are identically timestamped.

To support the process of Projection Point discovery, the Context-Broker keeps a database of previous user locations which can be searched by user, space and timestamp to return any location updates from other spaces, for the user concerned, that have been sampled co-temporally with the current update. This introduces the requirement that all user location updates submitted to the Context-Broker bear the timestamp which indicates when they were sampled.

Inputs to the Model

Our Context-Broker and model are designed to support application developers in a range of scenarios. These include:

- a developer who wishes to use an evolving context representation, such as location encoded by the set of Wi-Fi networks visible to the user, but allows another application to be responsible for maintaining and updating the information that describes the representation.
- a developer who wishes to introduce a second representation of context to their application, but needs support in reasoning about sensed information regarding the relationship of two context representations and the context of those using them.
- a developer who wishes to integrate the additional context information sensed by another application into his application, but needs support in reasoning about the relationship between that additional information and the context of his users.
- developers of separate applications who wish to combine their applications to allow their users to interact.

Several of the examples above feature collaboration between applications in order to access context-representations which may be complex or evolving or incorporate additional resources (Producers from other systems) into an application.

Just as a context representation used by an application developer can be created and maintained by a third party, so too can a user's sampled state within that representation. Behavio, for example, performs the sensing and gathering of multiple streams of user state information such as current environmental temperature, weather and location and makes this available for other applications to use as context. The Context-Broker supports this sort of distribution in the sensing, collection and construction of user states and context-representation models, by allowing any application to create or provide maintenance updates for a model space and by allowing any application to insert a user into that space or provide state updates for an existing user.

The tuplespace, known as Equip [GREENHALGH'02] , allows its clients to publish information structured as tuples of objects and subscribe to particular tuple patterns. When a tuple of objects has a structure that matches one of a client's subscription patterns, that object is delivered to the client with a notification indicating whether it is a new tuple or an updated version of an existing tuple. When the tuple is removed from the dataspace, the clients whose subscription patterns match its structure receive notification of its deletion. All objects inserted into the tuplespace as members of a tuple can implement matching algorithms, allowing patterns to match individual objects on their value or under other conditions such as subsets of the values of their class members. This flexibility allows the tuplespace to act as both a powerful filter and an aggregator for streams of information from disparate sources about representations and user states, combining and funnelling them to the appropriate modules of the context-broker for processing. When the context-broker publishes customised producer payloads and awareness values for a particular user back to the tuplespace, these tuples match the subscriptions of any tuplespace client collecting payloads on behalf of the user and those clients can collect and further filter or manipulate these payloads concurrently to each other.

Accessing the tuplespace through the API

Applications that engage the SPACES Context-Broker must be able to provide it with the appropriate information about their context-representation spaces

and users. Since the Equip delivers this information to the Broker via subscriptions to tuple patterns, the developer's API provides not only the interfaces that encapsulate context representations and user state in the structures of the model, but a platform for accessing the tuplespace using the appropriate patterns of publication and subscription. A Dataspace Proxy class provides methods for connecting to an Equip tuplespace, publishing Space and User information into it and receiving Space, User and Custom Payload information from it.

In our first implementation of the API, we focused on supporting those applications who construct and maintain their own Context-Representations either alone or in collaboration with other spaces. Scenarios that fit this pattern include those of the developer who wishes to introduce a second representation of context to his application, but requires help to reason about it, and the developers who wish to combine their applications, which use different representations of context, so that their users can interact with one another. Each context representation that an application provides a spatial specification for is managed in the API by a single DataspaceConnector object itself implemented from the Equip API. This object supports the publishing of information about the representation and of Producers and Consumers who inhabit the space. It also collects payload information addressed to Consumers that inhabit the space and passes this back to the application. The Dataspace connector is initialised by providing it with a reference to a Manager object that will handle responses from the Context-Broker, and with the uniquely identifying name of the representation's spatial model. All updates that are published to the tuplespace by the Dataspace connector contain this identifier and it is used by the Context-Broker to construct subscriptions so that all tuples bearing it are funnelled to those of its modules that manage and reason about the model of that particular representation.

Publishing structural information

A `setSpace(…);` method specifies the structures of the space that the DataspaceConnector concerns itself with and that the Context-Broker will be

responsible for modelling and reasoning about. Through this method the connector is passed the Space object itself, which contains the DistanceMetric, PointSet, and Aura Manipulation functions; the Space's BoundingBox object; and the separately URL from which these classes and all of their application-specific members can be loaded. These are published in a tuple that also contains the space's unique name.

Publishing User Information

User state information in the model is applied to the calculation of awareness, the creation of proxy presences and the discovery of Projection Point Pairings. The first two processes require only the location and aura of a user, but the latter involves comparing the sampling time of a user's previously received location updates from all spaces with her most recent location update. To this end, all user state updates published into the tuplespace must include a timestamp that represents the time at which the state was sampled or gathered.

The model splits users into two separate types, Consumers and Producers and as the API's spatial and user structures do not provide Consumer and Producer wrapper classes, it is the DataspaceConnector which maintains this distinction. A new Consumer is added to the tuplespace via a method which takes a Point representing the Consumer's location in the space, a Focus representing their area of interest, a text string representing their uniquely identifying name within the model, and a text string representing timestamp for the location update. Likewise, a Producer is added via a method which takes a Point, Nimbus representing their area of influence, unique name, timestamp, and a Payload object which encapsulates the information, services or behaviours that the Producer provides to Consumers with awareness of her. The name for each user is not simply unique within the scope of the space, but across all spaces in the model that the Context-Broker maintains. A group of application developers, each of whose applications operate within a different context-representation space have multiple ways to ensure that their users are identified uniquely across those spaces, the oAuth and openID platforms amongst them.

To update a user's state the Dataspace connector provides an `updateConsumer(…);` method and an `updateProducer(…);` method, both of which accept the same arguments as the add methods.

Receiving Payloads

The Context-Broker implements the model by reasoning about awareness between Consumers and Producers, and using the calculated awareness that a Consumer has of a Producer to obtain customised bundles of information, services or behaviours from the Producer. These Payloads must then be delivered to the Consumer, and to do this the Broker publishes them back to the tuplespace with the unique name of the Consumer they are intended for. A subscription to the tuplespace will generate three different types of event that the subscriber must handle, these indicate the addition, modification, and deletion of tuples from the tuplespace. When a Consumer achieves awareness of a Producer in the model, the Context-Broker publishes a tuple containing the Payload triggering an add event. As the Consumer's awareness level or type subsequently changes, the Context-Broker issues the result of these changes as updates to the tuple, triggering modification events, until finally, when the Consumer loses awareness of the Producer, the Broker deletes the tuple from the tuplespace triggering a deletion event. Each Payload tuple is labelled with the unique identifier of the Consumer it is intended for, allowing the Dataspace connector to place a subscription to receive all Payload tuple additions, updates and deletion events from the tuplespace, and thus the Context-Broker, for a specific Consumer. The Dataspace connector does this for every Consumer for which it publishes state updates.

The Manager interface in the API, which is used in the initialisation of the Dataspace connector, provides methods for handling these addition, modification and deletion Payload tuple events, and it is these methods which are called by the Dataspace connector whenever a relevant tuple event is received.

If a change to the model's state triggers a reassessment of a Consumer's awareness, it may result in changes to the awareness of multiple Producers for that Consumer. For applications that treat the individual Producer payloads

intended for a Consumer separately, the tuplespace Payload events triggered can be dealt with as they arrive at the Manager interface. However, some application may treat the awareness that a Consumer has of Producers in aggregate, for example, the voucher cloud application allows a user to search for coupons and special offers for restaurants and shops local to their current location. Using our Context-Broker service to reason about which vouchers to display, the application would submit each shop to the Broker as a Producer with a Payload of vouchers and each user as a Consumer whose location is a Point within a location representation, e.g. GPS, and whose Focus represents the distance within which shops should be considered local to them. When a user searches for vouchers, the application would probably prefer to collect all available voucher results and then order them and display the list to the user, rather than displaying them as their Payload tuples arrive.

This raises the question of how, given a set of Payloads currently published for a particular Consumer by the Context-Broker, an application can tell that the set of Payloads is wholly derived from the current state of the model and not representative of some intermediate state where updates are still being made to the Consumer's awareness in response to the most recent change to the model. To allow developers to make this determination clearly, the Context-Broker publishes a tuple for each Consumer that acts as a flag for the state of the Consumer's awarenesses in the model. This flag is identified by the Consumer's unique name and carries a single Boolean value. The value indicates whether the set of Payload tuples currently published for the Consumer represent the Consumer's stable awareness with respect to the current state of the model, or whether that awareness is still being constructed in response to the latest change to the model.

The API's Manager interface provides a method that takes a Consumer ID and the value of a flag tuple. The method is called by the Dataspace connector every time it receives a change in a flag tuple for any of the Consumers for whom it publishes state updates. Developers can implement this method to handle flag events for their Consumers.

Mapping the model to the Context-Broker

Our model approaches reasoning about user context by mapping context-representations to spatial structures and the values of user context to those of presence and awareness within these structures. The process of reasoning about context within and across context representations is thus mapped to reasoning about user awareness within and across spaces. To maintain this mapping of context to space, presence and awareness, a change in either a user's context, or in our understanding of the context's representation, must be reflected by changes to the spatial model which ultimately impact on user awareness. In our model, these changes can take the form of alterations to the space itself, for example adding a new cell tower to a GSM representation; updates to the state of a Producer, for example increasing her area of influence (Nimbus); to the state of a Consumer, for example moving from one GPS co-ordinate to another; or a change to the relationship between two spaces, for example the addition of new information indicating that the Wi-Fi location A90eF214 represents the same location as the Bluetooth location 76C21a92. The following sections explain the algorithmic behaviour that these updates must trigger in order that the model be maintained, and discusses the implementation of these algorithms in our Context-Broker system. We also provide a survey of the structures required to maintain the model in the Context-Broker.

Model Structures in the Context-Broker

presence and Projected presence

Every user exists in the model through her presence and projected presences. When a user has a point of presence in a space which derives from an application input to the Context-Broker that presence is called her native presence. We differentiate this structure from that of her proxy or projected presence because our model differentiates between the two in the way that it calculates Awareness. A projected presence, or a proxy, is not sensed or gathered by an application, but constructed within the model by reasoning about the location of the user's native presence, one that is sensed or gathered by applications, in relation to the projection point pairings that maintain relationships between different spaces. The Context-Broker maintains for every

space that it reasons about a record of that space's native users. This record is split further into a list of Consumers and one of Producers. As part of the structure representing a user, the Context-Broker maintains a list of that user's proxies in other spaces. The Consumer structures also store the user's native Aura, and the Producer structure stores the Producer's payload. The Consumer structures store details of the Consumer's awareness in this space and across spaces, they also store the customised payloads that accompany these awarenesses and they are responsible for publishing those payloads into the tuplespace for applications to retrieve.

Awareness

A Consumer's Awareness defines the relationship between her and every other Producer in the model and consists of a qualitive value –type and a quantitative value--level. Our model calculates Awareness using different procedures according to whether the Consumer and Producer are native to the same space or not. We call an Awareness that is calculated between a Consumer and Producer native to the same space a *Simple Awareness*. Awareness between a Consumer and Producer who are native to different spaces cannot be calculated directly, instead the model employs their proxy presences to generate the Consumer's awareness of the Producer in each space and then combines these two awarenesses to create what we call an *Aggregate Awareness*. Where awareness is calculated in the Consumer's native space between her native presence and a Producer's proxy presence, we call the awareness the Consumer's *Native Awareness* of the Producer. Awareness that is calculated in the Producer's native space between the Consumer's projected presence and the Producer's native presence we term *Proxy Awareness*.

In each Consumer structure, the Context-Broker maintains a list of Simple Awareness values which represent the user's current awareness of all Producers also native to her native space, a list of Aggregate Awareness values which represent her current awareness of all Producers in other spaces of the model, and a list of Native awarenesses which represent her current awareness of all Producer presences projected into her native space. In each Consumer

Proxy object within the Consumer structure, the Context-Broker maintains a list of Proxy Awarenesses which represents the awareness of the Consumer proxy with each Producer native to the space into which the Consumer's proxy is projected.

Space

Our implementation located the model structures that define a space in the API rather than the Context-Broker itself, but the employment of those structures in reasoning about context still takes place within the Broker and it is the Broker's SpaceManager classes which hold the code that performs this reasoning. Reflecting the different processes that are employed by the model to reason about awareness in different types of space, a Manager can be of four types: Continuous -Symmetric, Continuous-Asymmetric, Discrete-Asymmetric, and Discrete-Symmetric. Each Manager holds the structures that define the space upon which it reasons, and the lists of Producers and Consumers with presence in each space, which the Context-Broker maintains, are located in the space's Manager.

Projection Point Pairings

In the model, projection points hold the relationship between spaces and exist as pairings of PointSet members. We call each point in the pair an endpoint of the projection, and if we are speaking of the relationship from the perspective of a particular space, we call that space the native space, the endpoint located in that space the native endpoint, the space that the pairing links to the target space and the endpoint located in that space the target endpoint. In the Context-Broker the relationships between spaces exist for each native space as a list of the endpoints in that space paired with their respective target points. In this way Projection Point Pairings do exist as a single structure but as two structures one located in each of the spaces that the pairing links.

Model Processes in the Context-Broker

When spatial structure changes

A context-representation's spatial structure is described by its Distance Metric, Point Set, I-function and O-function. Dependent on these structures are the

states of its native users, the awarenesses of its native Consumers, its collection of projection points and therefore its relationship to other spaces, the states of its proxy presences, their awarenesses, the states of the proxy presences of all its native users and the foreign awarenesses of the proxy presences of all its Consumers. Where Consumers and/or Producers have proxy presences, the foreign awareness, one that a Consumer's proxy has of a Producer native to the space the proxy is projected into, is aggregated with the native awareness, that which the Consumer has in her native space of the Producer's proxy, to producer the Consumer's aggregate awareness of the Producer. The proxy and aggregate awareness of Consumers from other spaces, which have a relationship to the changed space, is also dependent on the structures of the space.

The first action when an element of a Space's spatial structure is altered must be to update the instance of the object that represents that structure. As detailed in the previous sections on model structures in the API and in the Context-Broker, the structures that define a space are encapsulated by the Space object. This object is held on the Context-Broker in the Manager responsible for that Space. An old spatial structure, for example a Distance Metric, can either be replaced directly, or replaced by replacing the entire Space object. The only spatial structures which exist in multiple places across a model are Points and Bounding Boxes which exist in multiple instances both within their own spaces' Managers, as user locations and as the native endpoints of projection pairings, and within the Managers of other spaces as the target endpoints of projection pairings. In the event of alteration to the structure, i.e. the class definition, of Points or Bounding Boxes within the space, every user and proxy location, and every projection point in the space, in addition to every projection point in other spaces with any relation to the space, must be updated to use the new location type. Text encodings of Points also exist in the database which our Context-Broker uses to store previous user locations and any change to the structure of a Point in the space would require altering all of those encodings. It is for these reasons that our implementation does not permit a change in structure of either a space's Points or Bounding Boxes.

Since the Context-Broker employs class loading, the spatial structures supplied by applications can change either in their class or in their state. Whilst a change in state necessitates only replacing the relevant object, a change in class definition requires the Context-Broker to load the new class before replacing the old object with an instance of it. Due to issues with the loading and unloading of classes in the JVM, where a spatial structure's class definition changes, the structure must use a new class name to ensure that the old class definition is no longer accessed.

Changing the I-Function

If a space's I-function, which manipulates neutral Aura entering the space, is altered, this may change the Aura size of all the proxy presences projected into the space. That change in Aura affects the awareness native Consumers have of proxy Producers and the awareness proxy Consumers have of native Producers. Where proxy Consumer awareness changes, this will impact upon the aggregated awareness that those Consumers from other spaces have of this space's Producers. When an I-Function change is received, first the old function, which is located in the Space definition within the Manager object for that space, must be replaced. Figure 27272620 illustrates the algorithms that must be enacted in response to the change and the spatial structures within which those processes take place.

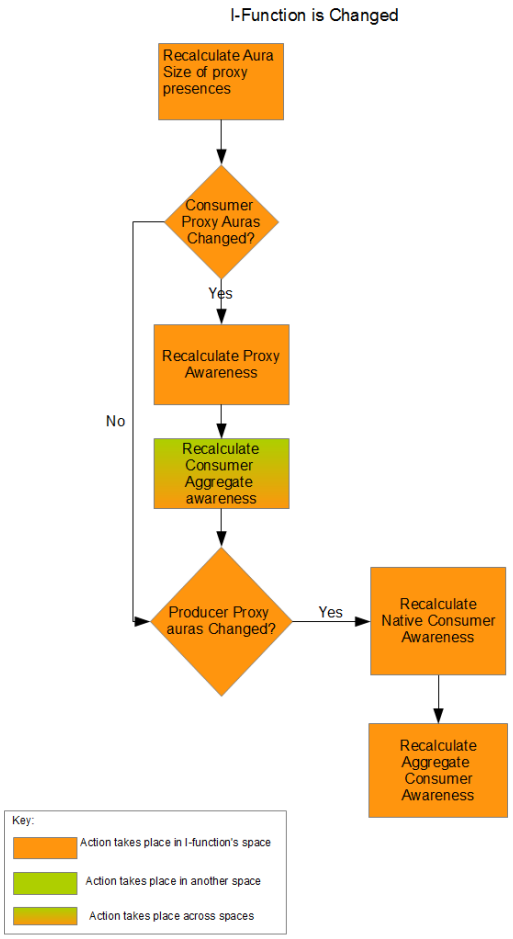

Figure 27 – processes involved in changing the space's I-Function

Changing the O-Function

Likewise, altering a space's O-function may change the Aura size of any user presences that the space projects into other spaces. Where the Aura of the presences projected out of the space change, the awareness of those Consumer proxies in other spaces will change and this will alter the aggregated awareness of the space's native Consumers. Since the change may also alter the Nimbus size of any Producer's projected presence, it can also cause a change in the awareness that other spaces' native Consumers have of projected Producers, and thus alter the aggregated awareness of the Consumers in those spaces. Figure 28282721 details the algorithms that must be enacted to update the model with respect to this change.

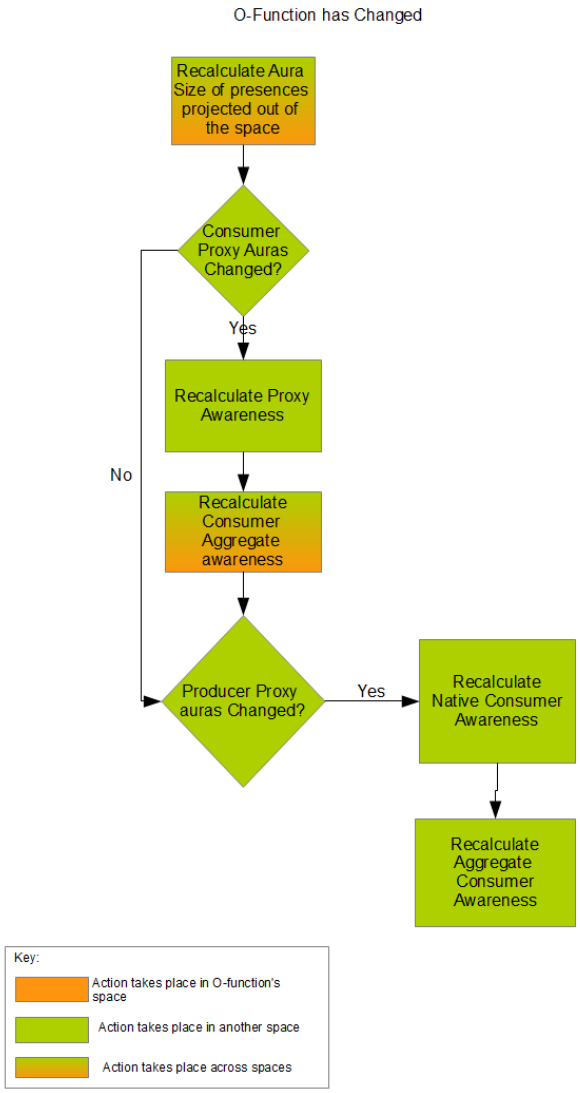

Figure 28 – Processes involved in changing the space's O-Function

Changing the Distance Metric

Altering a space's distance metric may alter the distance between native Consumers and Producers, causing their awareness of each other to change, it may also alter the distance between native users and projection point pairings. This can affect the existence of a user's projections, either by creating them where a projection point moves into the range of a user's Aura, or destroying them where the opposite occurs. In a less extreme manner, it can alter the state of a user projection, for example if a user Carrie has a proxy through projection point pairing P, and a change in the Distance Metric brings another pairing Q closer to Carrie's location, Carrie's projection must now be recreated through

Q. This will change the location and Aura size of the projected presence in the space it is projected into. If Carrie has a proxy through a projection point pairing which a change Distance Metric moves slightly closer to her location, her proxy's Aura size must be recalculated since the adjusted size of her Aura as supplied to the O-Function has changed.

If a change to the distance metric alters the distance between users and projection point pairings, it will also be altering the distance between native users and the proxy presences projected through those pairings from other spaces. Where the proxies are Producers, this necessitates the recalculation of the native awareness and aggregate awareness of the Consumers in the space, where the proxies are Consumers, it necessitates the recalculation of the proxy awareness and aggregate awareness of Consumers from other spaces.

Additionally, our model specifies that where a user's proxy presence is projected onto a Bounding Box, i.e. where the endpoint of a Projection Point Pairing in the target space is a Bounding Box, the proxy's location should be taken not as the Box itself, but as the point at the centre of the Box. Accordingly, we require that Bounding Boxes provide a method for retrieving that point, but the implementation details of this method are left to the discretion of the developer, who may choose to employ the space's Distance Metric in the calculation. Consequently, a change in a space's Distance Metric, even if it does not represent a change to the distance between a user and a projection point, must trigger an update to the location of any presence projected into the space onto a Bounding Box to ensure that this location remains the centre of the box. Since this is effectively a change in the proxy's location, this must trigger an update to the partial and aggregate awarenesses associated with the proxy - in the case of a Consumer proxy the proxy and aggregate awareness of the Consumer herself, in the case of a Producer proxy, the native and aggregate awarenesses that Consumers native to the space may have of the Producer. Figure 29292822 illustrates the algorithms that must be employed to update the model in the event of an update to the Distance Metric.

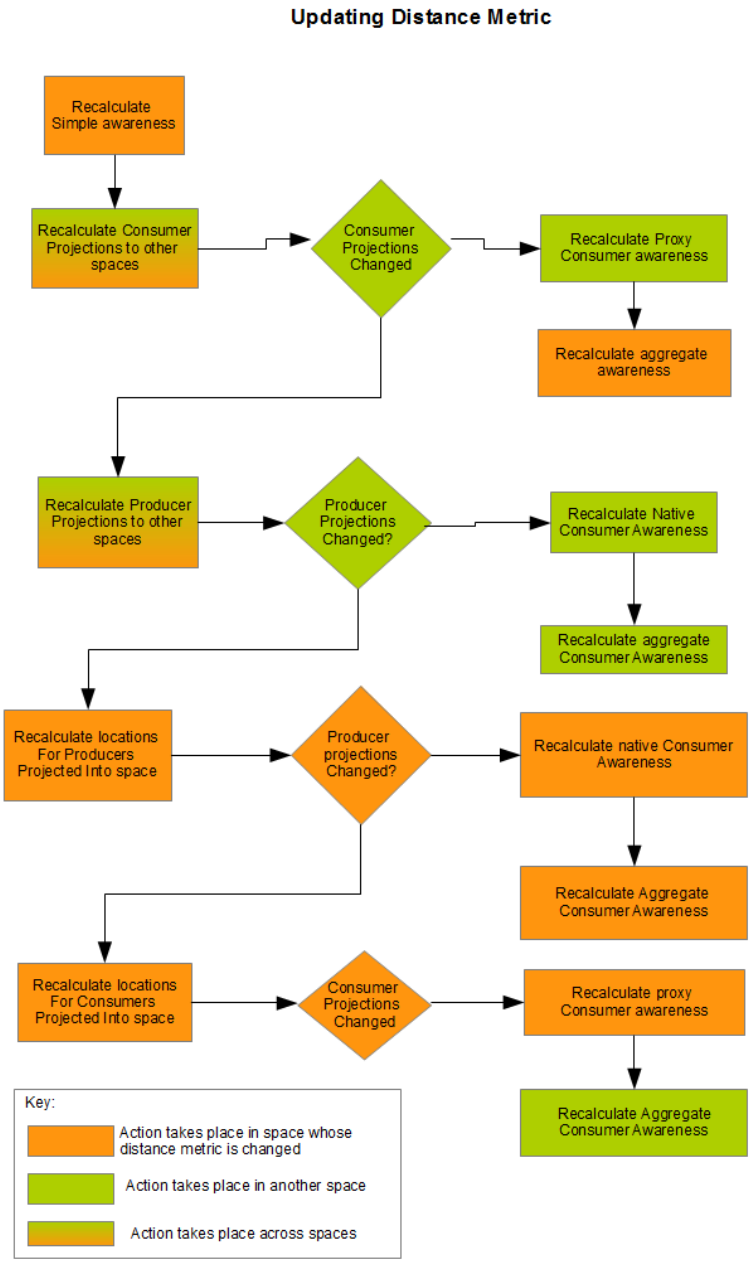

Figure 29 – Processes Involved in changing the space's Distance Metric

Changing the Point Set

Updates to a space's Point Set may take place through the addition or removal of its points. This change may affect the space's projection point pairings, either their existence or their value, and the validity of native user states. If a projection point pairing between one space and another has an endpoint which is a single point, rather than collection of points --a Bounding Box, and that

point is removed from the space's Point Set, the pairing is no longer a valid relationship between the two spaces and must be removed from the model. This removal can change the state or existence of any user proxies which have been projected into or out of the space via the pairing. If a projection point pairing has an endpoint which is a bounding box and one of the points contained in the box is removed from the Point Set, the Box must be updated to reflect this removal. This may lead to a change in the calculated centre of the box, or it may lead to a change in the calculated distance of the box from other points in the model. If the box's centre changes, the location of any user presence projected onto it also changes, which in turn affects the awareness that exists between native Consumers and proxy Producers and the proxy and aggregate awareness that Consumers from other spaces have of Producers native to the space. If the box's distance to points in the space changes this may cause the deletion of the proxy presence of any user projected through it, or the transferal of that proxy it to projection through a different, now closer, projection point pairing.

Where a Point Set has structure in addition to members and the space's Distance Metric acts upon that structure in calculating the distance between two points, the addition of new points or a change to the structure of the Point Set can produce changes in the space's distance metric. For example: Imagine a path A->B->C->D represents the shortest distance from point A to point D in an unweighted, undirected graph. The space's distance metric constructed to provide distances by walking the paths between vertices would give the distance between A and D as 3. If a new vertex Z is added to the graph with edges A-Z and Z-D, this introduces a new path from A to D, through Z, of length 2, this despite the fact that the Distance Metric's implementation remains unchanged. This means that any change to a Point Set with structure must be treated also as a change to the space's Distance Metric.

As illustrated by the scenarios above, when any aspect of a context-representation's model space is altered, it sets in motion a complex chain of checks and updates. In order to simplify this chain in our first implementation of the model, we concerned ourselves only with supporting evolving spaces

rather than volatile ones. That is, we support spaces only where information about the space is allowed to accrue over time rather than accrue and be deleted. Since a space's Distance Metric is required to map all pairs of points in its Point Set to a scalar value, even if that value is infinity, and likewise the I- and O-functions are required to return useable Aura transformations on all points in the Point Set, this restriction primarily prevents developers from removing points from a space's Point Set, and deleting projection point pairings. Since any effects that adding Points or structural elements (edges) to a Point Set may have on the model are enacted through the changes those additions make to the space's Distance Metric, the Context-Broker responds to updates to a space's Point Set in the same way as it responds to updates to the space's Distance Metric.

When a Consumer's state changes

In our model of context-aware applications, consumers draw to themselves the information, services, or behaviours relevant to them, based on their state, which comprises of their location and their Focus. When a Consumer's location changes it affects the distance between them and other users in the model, either directly, or because it affects the distance between them and any projection point pairings in their native space, which consequently affects the pairings used to project her presence into other spaces. When a Consumer's Focus changes, it alters the scale upon which she acts in the model, specifically her awareness of other users across the model and the set of projection point pairings that are considered local to her within her own space.

A change in the Consumer's location requires recalculation of her simple awareness of the Producers in her native space because it can alter the distance between them and her. This same change can alter the distance between her and the Producers projected into her native space, requiring the recalculation of her native awareness of their proxy presences, and the eventual recalculation of her aggregated awareness of Producers in other spaces.

Location change also affects both the existence and the state of a Consumer's proxies. Imagine a Consumer, Carrie, has a proxy presence which is projected into another space via a projection point pairing labelled P. If the change in

Carrie's location causes P to come to lie outside her Focus, and no other suitable projection pairing exists with her Focus to project her presence through, Carrie's projected presence will be deleted from the model. If Carrie's location then changes again, and this movement brings P back within her Focus, she will regain her proxy presence in the space that P maps to. If Carrie were now to move so that her distance from P was changed, but P remained within her Focus, her proxy's Focus may be altered by the change in the ratio of her native Focus size to her distance from P. Imagine one final change of Carrie's location brings the projection point pairing Q within her Focus and brings Q closer to her location than P. This will cause a change in her proxy's location and Focus size as it is now projected into the target space through pairing Q, which is a different distance from Carrie's location and links to a different endpoint in the other space. Figure 30302923 illustrates this algorithm and the modules of the Context-Broker in which it takes place.

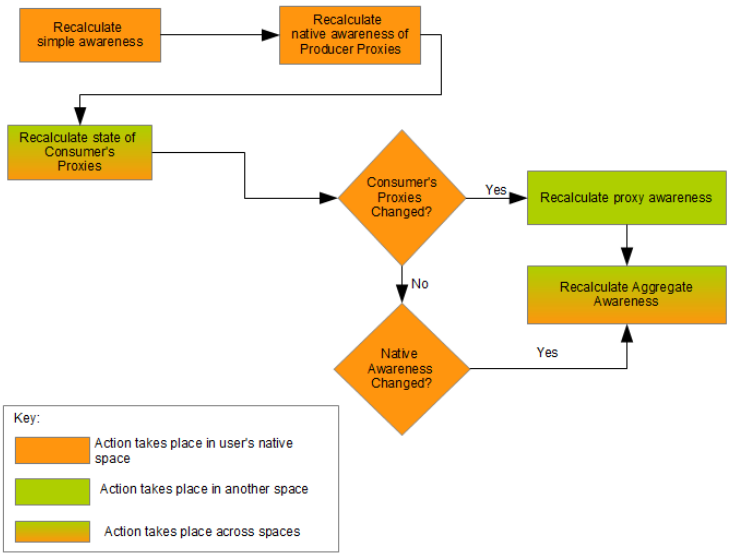

Figure 30 – Processes involved in a change to a Consumer's State

When a Consumer's Focus changes, it again necessitates the recalculation of her awareness for Producers native to her space, and for those Producers

projected into her space from other spaces, and consequently her aggregated awareness for those Producers. It also affects which projection points, if any, are considered local to her and this can affect both the existence and the state of her proxies, although in different ways to changes in her location.

If a Consumer doesn't already have a proxy in a space and her Focus increases in size to encompass a projection point pairing which maps to that space, she will gain a projected presence in the space via the pairing. Note though, that increasing her Focus where she already has a presence projected into a foreign space cannot, unlike changing her location, alter the existence of that projected presence, because by the definition of our model there must already exist a projection point pairing closer to her than the pairing which her enlarged Focus now encompasses. If the Consumer already has a proxy presence in a foreign space, and her Focus is reduced in size so that it no longer encompasses the projection point the proxy is projected through, that projected presence will be removed from the model. Again, the presence cannot be transferred to another projection point pairing as is possible with a change of location, because by the rules of our model there are no other projection point pairings in that particular foreign space closer to the user than that currently being used as her proxy. Additional to their creation or destruction, a change in the Consumer's Focus size can alter the proxy Focus size for any projected presence she has, because they are related to the ratio between the distance from her location to the projection pairing and the size of her native Focus. Figure 31313024 shows this algorithm and the modules of the Context-Broker in which it takes place.

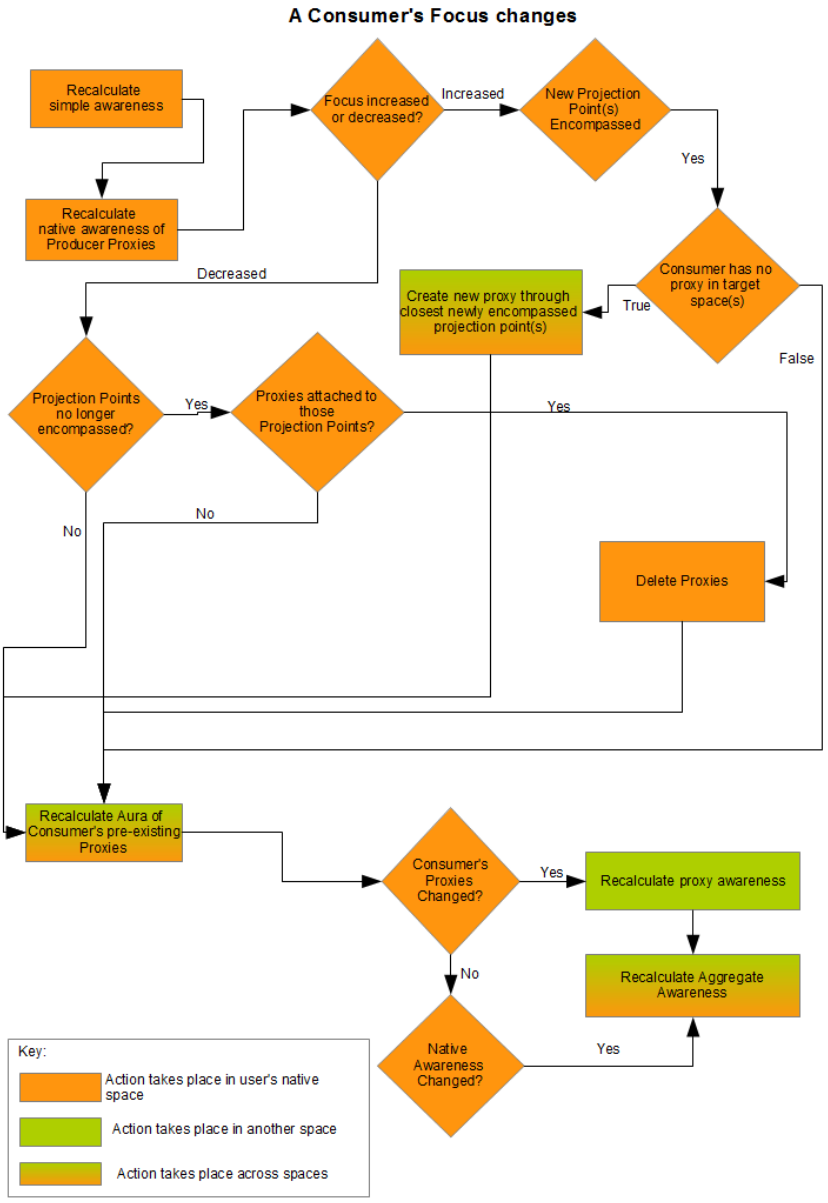

Figure 31 – Processes involved when a Consumer's Focus changes

The scenarios above suggest that when a Consumer's state changes in the model the Context-Broker must perform checks for the following conditions: changes to the Consumer's awareness of Producers native to her space, changes to her awareness of Producers projected into her space, changes to her proxies, which leads to changes to the awareness her proxies have of Producers native to the spaces they are projected into, and changes to her aggregate awareness. The scenarios also suggest some shortcuts that can be taken. For example, if

the only change to the user's state is that her Focus has decreased in size, the Context-Broker does not need to check its projection point records for mappings to spaces that she is not already projected into. If an increase in the user's Focus is the only change to her state, the Context-Broker needs to recalculate the Focus size of her existing presence projections, but does not need to change the projection point pairings that they use and thus does not need to recalculate their location. Figure 31313024 illustrates these procedures in a flowchart structure.

When a Producer's state changes

Our model presents Producers as sources of Information, Services or Behaviours, which we call payloads. Producers customise these payloads for Consumers based on each Consumer's awareness of the Producer. When a Producer's state changes, it is either due to a change in her location, her Nimbus, or a change in her Payload.

Producer Payloads, by our model's definition, may represent not only data, but services and behaviours. Moreover, in our API's definition, Payloads also encapsulate the method by which their data, services, or behaviours are customised before delivery dependent on a Consumer's awareness of the Producer. A change in Payload, unlike a change in a user's Location or Focus can therefore be either a change to the data carried by the Payload or a change in the Payload's class definition. Since all Producer payloads are implemented by developers as subtypes of the Payload class, a developer can change the behaviour of a payload by replacing the payload object with one of a different Payload subclass. When a Payload's content changes, but the Producer's state otherwise remains the same, the Context-Broker then must regenerate each of the payloads that the Producer currently supplies to Consumers. To support this, where awareness exists, the Context-Broker's user management modules store the current awareness level that each Consumer has of each Producer.

When a Producer's Location changes, the effects on the model are much like those involved in changing a Consumer's location, but rather than refiguring the awareness that one Consumer has of multiple Producers, the awareness that

multiple Consumers have of the Producer is what must be adjusted. Thus, the awareness that Consumers native to the Producer's space have of the Producer must be recalculated, as must the proxy awareness of Consumers projected into the Producer's space with respect to the Producer. A change in location also affects the distance between the Producer and any projection point pairings in the space. This can affect the existence and state of the Producer's proxies in the same way that it affects the existence and state of the Consumer's proxies. These changes to the Producer's proxies may change the native awareness that Consumers in other spaces have of Producer presence images and in turn the aggregated awareness that those Consumers have of the Producer.

When a Producer's Nimbus changes, like the Consumer with her Focus, the scale upon which the Producer acts in the model is changed. This affects the simple awareness that Consumers native to her space have of her, the proxy awareness that those Consumers projected into her space have of her and therefore their aggregated awareness of her. It also affects which projection point pairings, if any, are considered local to her and this affects the state of her proxies in exactly the same manner as a change in Focus size affects the state of a Consumer's proxies. Any effect on her proxies can affect the native awareness that Consumers in other spaces have of her, and therefore their aggregate awareness of her.

The scenarios above suggest that a change in a Producer's state must trigger the Context-Broker to check for the following conditions: For the Producer herself: changes to proxy presences with the same restrictions and optimisations used for handling changes to Consumer state; for Consumers native to the Producer's space, changes to simple awareness; for Consumers native to other spaces with presences projected into the Producer's native space, changes to proxy awareness and changes to aggregate awareness; for Consumers native to spaces that the Producer is projected into, assuming changes to the projected presence in that space, changes to native awareness and changes to aggregate awareness.

When the relationship between spaces changes

Our model encodes the relationship between spaces as projection point pairings, which are pairs of either points or contiguous sets of points, which we call Bounding Boxes, whose pairing signifies that they occupy the same point in some aggregate space. The model uses these pairings to project user presences from one space into another, meaning changes in their state are impact on the existence and properties of all presence-images in the model and therefore to all Native, Proxy and Aggregate Awarenesses in the model, but not Simple Awareness values.

Since our first implementation of the model supports only spaces which evolve rather than those in which information about the space is volatile—can be removed—the Context-Broker performs two operations with respect to Projection Point Pairings: Creation, in which a new pairing is added to the model and Update, in which an existing pairing has additional points added to it. For example, let there be a projection point pairing P between the spaces GSM and GPS that comprises of two points: the node 83945 in the GSM space and the co-ordinate (0,0) in the GPS space. It is detected that the point (0,0.001) also maps to the GSM node, so the projection point pairing P={83945, (0,0) } is then updated to include that co-ordinate. This changes P into a pairing of a GSM point and a GPS Bounding Box consisting of the points from (0,0) to (0, 0.001), P = {83945, [(0,0),(0,0.001)]}

To add further points to each Projection Point in pairing P we must be able to determine if a new linked pair of presence updates includes a Point which is a member of either of the Projection Points of P. For example, if we determined from co-located, co-temporal updates that the GSM node 83944 and the GPS coordinate (0,0) were the same point in some aggregate space, we would like to add the GSM node to the pairing. In order to do this, we must be able to ascertain if a point from a newly discovered pairing is a member of any existing pairing including as a member of any pairing's BoundingBox. It is, however, fairly inefficient to run through the membership of every BoundingBox involved in every Projection Point Pairing. In order to detect new Projection Point Pairings, we keep an SQL database of the point, space ID,

user ID and timestamp associated with every user update submitted to the Context-Broker. When a new user update arrives, it is logged in the database and checked against existing entries for updates from that user in other spaces which match the current update's timestamp. In addition to this, every point entity in the database may be linked to a BoundingBox id. When that point is subsumed by a BoundingBox in a Projection Point Pairing, it's entry in the database is linked to the Box's ID so that any further relationships concerning it can then be made against the Bounding Box rather than the Point itself.

Projection Point Pairings themselves are stored in a hashtable so that they can be accessed and updated easily. Given a newly detected pairing between two points A and B, we first check if either point is in the hashtable as an unwrapped member of a projection point pairing. If we find a match with A but not B, we can update the projection point pairing A → X so that it now reads A→{X, B} where {X, B} is a new Bounding Box containing the point X and B or where X was already a BoundingBox, an updated Bounding Box containing all the points of X and the point B. We then update the database entries for B so that we know that B is a member of the BoundingBox {X, B} (and for X, if X is a point not a BoundingBox). If we find a match with B but not A, we follow the same procedure, updating the mapping B→Y to B→{Y, A} etc.

If we find no match in the hashtable for either A or B we should check the database to see if either point is already wrapped with a BoundingBox. If we find only A is already wrapped in a BoundingBox, we should retrieve the Box's ID, find the relationship {A,…} →X in the hashtable, and follow the procedure for adding B to the relationship {A,…}→X as described above. If we find that only B is already wrapped in a BoundingBox, we should follow the same procedure to retrieve the projection point relationship from the hashtable and augment it with A.

If we find both A and B are both already members of Bounding Boxes we should first check if they are members of the Bounding Boxes in the same Projection Point relationship and if so we need do nothing as their relationship is already recorded in the model. If they are members of different Bounding Boxes so that we have two Projection Point pairings {A, …}→X and

Y→{B,…} where X and Y may be either Points or Bounding Boxes themselves, we need to combine these two pairings to give {A,Y,…}→{B,X,…}. If X and Y are both points this can be achieved by simply adding Y to the Box {A,…} and X to the Box {B,…}. If on the other hand X is a Bounding Box itself, we must call the `getPoints();` method on it in order to extract its component points and add each of these to the Box {A, …} updating their relations in the database to reflect the change in their BoundingBox ID.

If neither A nor B have relationships to a BoundingBox in the database, the detected relationship is entirely new and we must create a new Projection Point Pairing by adding A and B to the hashtable in the relationship A→B.

When a new Projection Point pairing between two spaces is added to the model, the Context-Broker must check if this alters the state of any image-presences for users native to the two spaces. Any user in the spaces whose Aura the new projection pairing's endpoints lie within may receive a new image-presence if they do not have one for the space already, or an update to an existing image if the new projection point is closer to their location than the pairing that they currently use. If any Producers in either of the spaces undergo changes to their image-presences, the Context-Broker must update the native awareness and aggregate awareness of Consumers in the space that the image projects into. If any Consumers in the spaces experience changes to their images their proxy awareness of all Producers native to the space they are projected into must be updated and following that, their aggregate awareness of those Producers

When a Projection Point is updated, a point is added to either endpoint of an existing projection pairing. For users native to the space, this can change their distance from the point which may bring it inside their aura or change their native and therefore aggregate awareness of any Producer projected onto its centre. For users already projected through that pairing, the update triggers a change in the image's Aura and in an extreme case may trigger the deletion of the image. Where a user is not already projected through a projection pairing, the addition of an extra point can bring the pairing's endpoint within the user's

Aura, and the Context-Broker then has to check whether this point is the closest to the user of all projection points within its Aura that map to that particular space. If it is, this will trigger the creation and projection of an image-presence through that pairing, or the update of an existing image-presence to use the new, more suitable projection point.

For image-presences, an update to the target Bounding Box of the pairing that they are projected onto can change the centre of that Box and thus their projected presence location. If this change occurs for a Producer image, the Context-Broker must recalculate the native and aggregate awareness levels for all Consumers in the space, if it occurs for a Consumer image the projected presence's proxy and aggregate awareness of the space's native Producers must be recalculated.

Again, we can see that all of these processes must trigger a recalculation of awareness. Where it is a Producer's image presence that has been created or whose location or Nimbus have changed, the recalculation is of the native and aggregate awareness of Consumers in the space the Producer is projected into. Where a Consumer's image undergoes the same changes, it is the Consumer's proxy awareness and aggregate awareness of the Producers native to the space she is projected into that the Context-Broker must recalculate.

Conclusion

The SPACES model, presented in Chapters 3 and 4 provides a methodology for approaching context generated by heterogeneous technologies as a form of spatial presence and awareness between users, and details mathematical algorithms for reasoning about that presence and awareness both within and across the spaces generated by different types of context technologies.

In this chapter we explored how the architecture of context-aware systems and the process of constructing models of context representations may affect our choices in implementing the model. We presented our implementation of the

model in the form of a Context-Broker service which is accessed by other applications via an API and tuplespace. This implementation splits responsibility for implementation of the model's structures between applications and the Context-Broker, requiring applications to implement the spatial structures: Space, Distance Metric, Point Set, Point, Bounding Box, I-Function, and O-Function and the user structure of Payloads. We detailed the features of those structures that are exposed to the Context-Broker and how they contribute to the maintenance of the model and the performance of presence and awareness reasoning. The structures of the model that are implemented by the Context-Broker we detailed as being those that exist across spaces-- the Consumers, Producers and Projection Point Pairings, and those that implement the reasoning processes of the model—the four different types of Space Manager.

We also discussed the way in which the use of a centralised Context-Broker service can support collaboration in the construction of spaces and described how the tuplespace upon which the Context-Broker is located supports this by allowing aggregation and filtering of streams of user and space information originating from multiple sources. We described how the Dataspace Connector of our API supports and structures the way in which applications access the tuplespace so that the information the Context-Broker receives is always appropriate.

One of our model's key aims is to support reasoning about context across its different representations, and its execution of this depends on detecting, building and maintaining information about the relationships between those representations, so we discussed the procedures some applications use to detect this information, and described the procedure employed by our Context-Broker in the detection of Projection Point Pairings. Finally, we explored the algorithms involved in the process of maintaining a model of multiple spaces populated by users and their projections by describing the procedures that the Context-Broker must follow in the event of various changes and inputs to the model.

Our aim in constructing the Context-Broker system is to provide developers with a platform upon which they can build context-aware applications and that will support those application in reasoning about user context both within and across its different representations. In the next chapter we explore the user of and issues with our system by using the SPACES platform to build and test a location-aware game whose context-reasoning is performed by the Context-Broker.

Glossary of New Terms

Bounding Box A programmatic construct used where a collection of points is involved in a projection point pairing to wrap and represent that collection. Depending on developer's implementation choices may be a true bounding box or simply a set of points.

Chapter 6

Using the SPACES Platform

Introduction

In previous chapters we discussed the properties of various types of context employed by context-aware applications. From examination of these properties, we were able to classify context representations into four different categories. Extracting the distinguishing features of these categories allowed us to model them as four types of space in which users reside. This approach to context representation-- as spaces populated by users who consume or produce information-- formed the foundation of a conceptual model that supports reasoning about user context as an awareness between producers and consumers that triggers the transmission from Producer to Consumer of customised Payloads of information, services or behaviours.

Our model supports reasoning about context between Producers and Consumers who use different sensing technologies and thus have different representations of context by allowing a user in one space to project an image of her presence into other spaces. This projection takes place wherever the model has information about the relationship between one space and another which is local to the user. The awareness between two users across spaces is then calculated as multiple awarenesses between their sensed and projected presences and reconciled to a single model-wide awareness using a process of awareness aggregation.

We presented this conceptual model, its structures and their features in Chapter 3. In Chapter 4, we explored the mathematical aspects of space, presence and awareness in the conceptual model, using these to construct the equations and calculations required to define and compute awareness mathematically under different circumstances within spaces and to aggregate awarenesses across spaces.

In Chapter 5 we presented SPACES, a realisation of our model's spatial structures in the form of a developer API and a Context reasoning service: A Context-Broker, that implements the computational procedures of our model. The Context-Broker service takes as its input information about the spatial structures of a context-representation encapsulated by the API and information about the state of users located within that space. It creates as output for each Consumer a set of customised producer Payloads derived from the Consumer's awareness of Producers across the model space.

We began this work with an aim to provide support for developers of context-aware applications by providing them with a simple way to think about context and removing the burden of reasoning about context and of reconciling the forms of context generated by different technologies to a third-party context-reasoning service. In this chapter we explore how our SPACES model and platform can be used to this effect. To do this we conceived of and implemented a mobile location-aware game which employs our Context-Broker and its API to allow users with GSM location-sensing technology and those with GPS location-sensing technology to play in the same space without the developer having to manage the relationship between the two sensing technologies. We describe a demonstration of this game that features a user active in one space accessing objects created by users in the other space. During the course of this demonstration, the user experienced problems with the application which we present and classify here.

Designing the application

What do Context-Aware applications do?

We define context-aware applications as those that adjust their behaviour based on information either sensed or gathered about a user's current state so that they can better support the user in her activities; this adjustment can take many forms. Tasker is an automation application for Android that allows users to specify behaviours to be performed when certain conditions are met. For example, a user who likes to remain undisturbed at work might specify that when his phone is both within 500m of his workplace and positioned face down, the ringer should be set to silent. Here the behaviour adjusted is how the

phone handles incoming calls and notifications and this is done automatically to support the user's goal of working without unnecessary interruption. Google's Search Cards application, aka Google Now [GOOGLE NOW, '15] , performs a similar job for user searches, allowing users to specify sets of context-specific information they are regularly interested in, for example the weather for their current location or film screenings at their nearest cinema. When the user accesses a card they are presented with the information as it applies to their current location, rather than having to enter that location manually. Hyper-local notifications system Roximity [ROXIMITY, '15] allows users to elect to receive notification of venue-specific special offers to a range of devices when they are in close proximity to a venue. In this case the behaviour adjusted is when the user receives a coupon, which is based upon their location.

iOS app Glyphics [GLYPHICS, '15] allows users to construct picture and text messages for their friends and pin them to a particular location or physical object by pointing their phone's camera at that object. Scanning the same area with their phone's camera, the user's Facebook friends are then able to see and retrieve these messages. The behaviours that are supported here are communication with friends and finding communications from friends. Similarly, Acrossair [SAHA'14] allows teachers and students to construct interactive tours by attaching to specific locations items such as text, photographs, and multimedia that can then be viewed by others.

Social discovery application Highlight presents its users with a list of people nearby who have some form of social connection to them, or with whom they share an interest, in order to support them in meeting new people and catching up with friends whilst out and about. Here the behaviour that is adjusted is the selection and ordering of people to display to a particular user. Couponing application Vouchercloud's display of lists of coupons and offers for nearby stores when a user accesses the application is a similar type of action [VOUCHERCLOUD, '15] .

The previously mentioned Acrossair allows content-creators to share their content in the form of information attached to real world objects, and since the

application is designed to support the creation of augmented reality tours and fieldtrips, the application can also be said to allow users to find out more about a particular location, or landmark. Other context-aware applications that support this function are Nokia's City View and AOL's Entrance [MLOT'12] . Both provide information about venues such as cinemas in a location-aware or Augmented Reality format to users who view the venues through their phone's camera. The Jewish Timejump game [JEWISH TIME JUMP: NEW YORK, '15] expands upon this approach by creating a location-aware gamified history tour of New York City in which students take on the role of a reporter investigating the events surrounding the Uprising of 20,000-- a worker's strike led by Jewish women in the garment industry for better pay and conditions. Users entering different locations across the gamespace, which spans large swathes of the city, can explore video, photographs and text information that give them information about the history of the Uprising as it relates to their current location.

Examining this selection of applications patterns of functioning begin to emerge and from these patterns we can say that context is normally used in four different ways: *Discovery*, *Exploration*, *Artefact Creation* and *Automation*.

What do Context-Aware Application Developers need from SPACES

Discovery functions allow users to find out what is near to them, or to their context if their context is not location. For example, users of similar interests can be said to be considered semantically near to each other. Exploration functions allow users to discover more information about a specific thing, for example a place, person, or landmark. Here context is used to select the thing to be explored, for example the user's location in Jewish Timejump selects the set of historical information they will explore. Artefact Creation allows users to annotate physical or virtual things with virtual objects that persist and can then be seen by others. From the aspect of a venue owner, the Vouchercloud and Roxio apps support just this behaviour, allowing them to create coupons which are then seen by users near to their location. The Glyphics iOS app is designed

around this sort of usage also. Finally, Automation functions allow applications to take responsibility for carrying out behaviours under certain contexts that users would ordinarily have to do themselves. Tasker is designed to support just such functions. For our platform to be useful to developers, it should support applications in performing each of these four functions, as well as providing accurate context reasoning within spaces and reasoning across spaces that is as accurate as the information about their relationship permits.

What do Context-Aware Application Users need from SPACES

Many of the applications above revolve around presenting some form of information to the user. For example, in Highlight the user is shown a list of the people nearby to whom she has some connection through mutual friends or shared interests. In Vouchercloud she is shown a list of coupons and offers for nearby venues, and in Acrossair, Glyphics and similar augmented reality apps, she is shown virtual objects embedded in real world landscapes. It is only in Tasker that the user does not have to make sense of information the application presents to her, instead she must trust the application to behave appropriately to the situation. What is appropriate behaviour here is determined by the user at an earlier time. In this aspect, usage of Tasker is similar to the usage of a content-creation application by a content creator --the user specifies that some behaviour should happen under particular circumstances and must trust for the application to detect those circumstances and perform the behaviour correctly based on them. In Glyphics the behaviour specified by the user is the display of a particular Glyph to other users who are looking at the location it has been pinned to and this specification is performed through the action of creating the Glyph and pinning it to the location. In the same way, silencing a phone's ringer when it is placed face down in a particular location is a behaviour specified by the user through programming the event trigger into Tasker. In order for users to find a platform useful they must find that it both supports them in understanding information for those aspects of context-aware applications that revolve around presenting information to the user and in understanding the context representation for those aspects that revolve around constructing application behaviours that are to take place in different contexts.

Designing the Bombsquad Game

To demonstrate our Context-Broker service in action, we built a location-aware game called Bombsquad that tasks users with identifying the location of virtual bombs in a physical gamespace based on clues left by the games master. Users interact during the game by leaving each other notes that are helpful or misleading in their quest to be the first to identify all the bombs.

Figure 32 - The game is played on a Nokia N90 phone. The user carries a GPS receiver (top right) to provide her with presence in the GPS space

The gamespace is established by a games master who uses a mapping application to plant both virtual bombs and clues to their locations around a physical landscape. Clues can be in the form of text, photographs, or video. Gameplay then proceeds as users with mobile devices move through that physical landscape performing searches on their device which return screens showing objects, which may be clues or mines, and notes from fellow players (Figure 33). Notes are coloured orange to indicate that it is safe to view them, but at the start of the game clues and bombs are both coloured yellow and the user does not know when she views a yellow object if she will get a clue or explode a bomb. Users begin with a health level of 100, but when a user explodes a bomb by viewing it, her health level decreases. If a user views an

object and it turns out not to be a bomb, it will give her a vital clue to the location of a bomb.

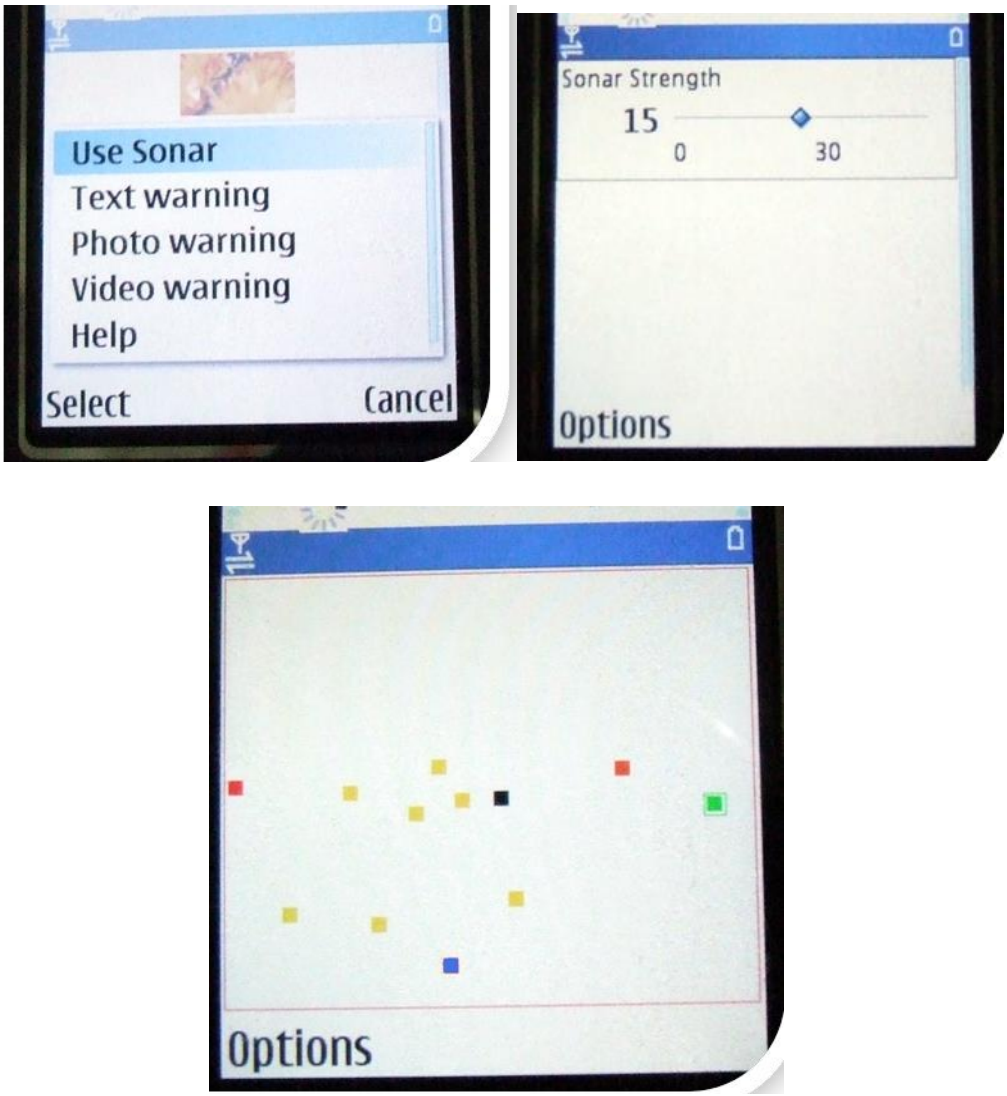

Figure 33 - The user searches for virtual objects and receives a map of the objects relative to her position

Clues which describe the locations of mines are delivered to the user scrambled and the degree of scrambling depends on the user's proximity to the clue (Figure 34). If a user believes a particular object is a bomb, she can mark it as such. When she has done so, wherever she is in the gamespace if that object appears in her search results, it will be coloured pink. Likewise, if she is unsure about an object she can mark it a blue, and if she is sure an object is safe, she can mark it green. This helps users to keep track of their opinions on objects as they move around the gamespace. The aim of the game is to be the first player to decode the clues and mark the location of every bomb without losing all of

your health, and when a user has correctly marked all of the bombs in the gamespace if she is the first to do so, she is declared the winner. A user who uses up all of her health by encountering too many bomb explosions is dead and automatically loses the game. If users wish to communicate, to collaborate or to mislead each other, they can create a note which can be either a video, text or a photo and upload it to the game server (Figure 35). The note is pinned in the gamespace to the location it is uploaded from and instantly becomes available to other users who perform searches in the note's vicinity. Notes can also be used as landmarks to help users navigate around the Mixed Space and we seeded the gamespace with some notes of this sort in our first test of the game.

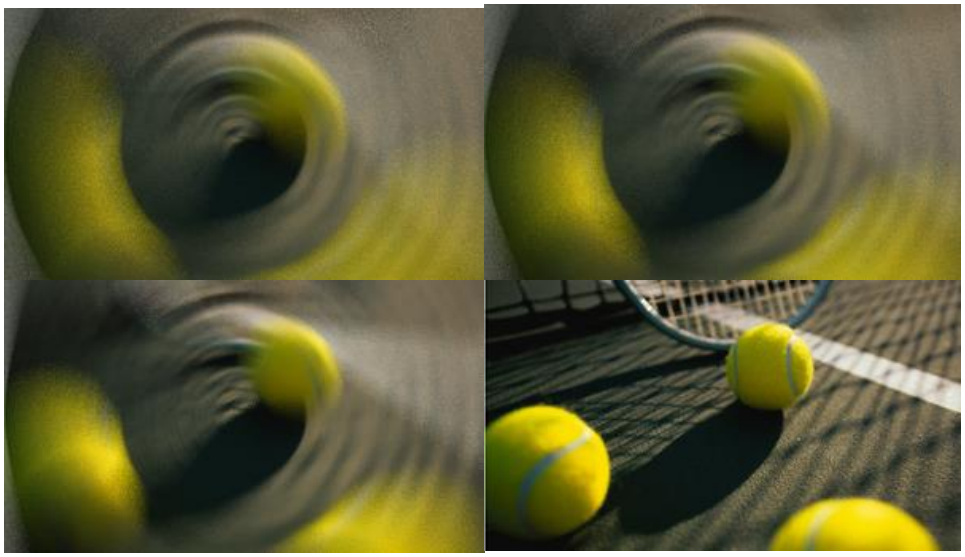

Figure 34 - Clues become progressively clearer as the user's awareness of them increases. This photo clue has four modes mapping to the four modes of qualitative awareness. The clearest image is delivered when the user has high awareness of the clue

Figure 35 - The user can place notes in the gamespace, for example to remind her where physical landmarks are

The game is a good test of the context-broker service's usefulness to users and developers because it performs three of the four functions that are frequently employed in context-aware applications. When users search for nearby objects, the application is enacting the discovery function. When they view a particular object, they are using their context to explore it - recall objects alter in their appearance or behaviour when viewed according to the proximity of the user viewing them. Additionally when users create notes for each other, they are undertaking artefact creation.

We did not explore the function of automation in this application due to the conditions under which the application was developed. The application was constructed to submit updates in user location to the server (and thus the context-broker) only when the user performed a search or accessed an object. This was because we felt the internet connection available to mobile devices at the time was not fast enough to support constant streaming of location updates. If we had had access to a faster mobile connection we could have, for example, added a feature to the application that consistently updates the user's position with the server and the list of objects she is in very close proximity to, and triggers a bomb explosion if the user passes very close to the bomb's physical location regardless of whether she views it or not, and this would have approximated a similar context-aware application behaviour archetype to that used in Tasker.

Implementation of the Game

The game is implemented as two systems, one GSM the other GPS oriented, and these are split into mobile client and web-based server applications. A third mobile client communicates with both servers representing users who have access to multiple context-sensing technologies.

The mobile applications

The mobile clients are implemented as J2ME applications on Nokia Symbian phones. In order to play the game, the user must first register with the server, which they do by providing a username. Each mobile client provides two activity choices for a user. Search and Create Note. The search option allows the user to search for objects specifying the size of search that should take place from small to large, this sets the size of their Focus for the search. When the user submits the search her location is sampled, using whichever technology the mobile client employs, and submitted to the application server along with her focus, ID, and a timestamp which identifies when her location was sampled. The client then waits for a response from the server which contains the objects found within the user's search radius, or more precisely the customised payloads of data or behaviours that those objects have generated based on what the Context-Broker calculates to be the user's awareness of them.

Any objects returned by the search are displayed to the user in a 2D layout on screen, from this she can select any particular object and either label it - bomb, safe, or unknown, or view it. Since each object has a unique ID these labels are remembered from search to search. The user can continue to move around the physical gamespace, searching for and labelling objects until she has either used up all of her life energy by detonating too many bombs or correctly labelled all bombs in the gamespace.

If the user wishes to at any point during gameplay, she can create a note. The mobile client application allows her to select the type of note she will create and once she has typed, recorded, or photographed her note she can set a Nimbus size for it. When she uploads the note to the server, the mobile

application samples her current location and this is submitted along with the note's contents and Nimbus.

When a user of the third type of mobile client, which represents users who have access to multiple context-sensing technologies, performs a search, her mobile client samples both her GPS and GSM location and submits one search request to each server. Both searches are submitted with the same timestamp which will allow the Context-Broker to deduce that the locations it receives in those Consumer state updates should be taken as part of a projection point pairing.

Implementing the model structures

The server applications are implemented as Tomcat applications using a mix of JSP and J2EE. To maintain gameplay, each application keeps track of where its users are, how much life energy they have and the labels each user has currently assigned to objects in the gamespace. Since a user can only win the game by correctly labelling all bombs in the gamespace, the applications must themselves know of all bombs in the gamespace. Since the model and the Context-Broker system do not provide a facility for an application using one space to retrieve all Producers or Consumers in another space, the two server applications guarantee that they know all of the bombs across the two gamespaces by sharing a list of the unique IDs which identify their bombs to the context-broker. When the application detects that a user has labelled all items on this list, and only those items, as bombs, the user wins the game.

The servers encapsulate their context-representations using the API of the Context-Broker's developer platform. As described previously, the GPS space's bounding box is defined in the model of the space as an axis-aligned square whose base is parallel to the equator (the line x=0 in the model space) that contains every point in the box. The GPS server application implements this by including 4 additional members X_1, X_2, Y_1 and Y_2 in its Bounding Box class, which represent respectively the coordinate values of the left, right, top and bottom of the box, and by customising the `addPoint(…);` method of its Bounding Box class so that when a point P is added

- If it is the first point added to the box all the coordinate pairs are set to the value of the point's coordinates.
- If it is not the first point to be added to the box
    - P's y-coordinate is compared with the y-coordinate of the bottom of the box, Y_2 and if it is less than Y_2, we replace the value of Y_2 with that of P's y-coordinate
    - Similarly, if P's y-coordinate is greater than the value of Y_1, and if P's x-coordinate is less than the value of X_1 or more than the value of X_2.

The Box also maintains an internal list of the individual points added to it which can be returned as the result for a call to its `getPoints(…);` method. To compute the centre of the Bounding Box the GPS implementation returns simply the centre of the square which is calculated as the coordinate values (X_1, Y_2) plus half the distance from X_1 to X_2 and Y_2 to Y_1, respectively.

The application implements the GPS space's Symmetric metric as a class which given two points computes the Euclidean distance between those coordinates. When given a Point and a Bounding Box, the implementation takes the distance between them as the distance between the Point and the centre of the Box. A GPS Point class has two members which are floating point x and y coordinates for the space. The space's PointSet class implements the abstract class ContinuousPointSet. It provides a single custom method which given a Point checks whether each of its co-ordinate component figures lies within the bounds of possible GPS readings.

The GSM space's Bounding box is modelled as the smallest subgraph, or group of unconnected subgraphs, that contains all the points identified as being part of the box. The server application implements this as a class that stores an array of the Points in the subgraphs. The `addPoint(…);` method of the class checks first that the point belongs to the space and then that it does not already belong to the box. Having ascertained both of these, it performs a greedy breadth-first search of the Pointset graph for the vertices that are members of the Box, originating from the point to be added. If that search concludes without finding any vertices, i.e. the Point being added is a member of a

disconnected subgraph of which none of the Box's vertices are members, the Point is simply added to the box. If a vertex is found the search concludes and the point and the vertices that comprise the path between the found vertex and the Point itself are added to the Box. We feel this approximates a sense of boundary well enough for a space that does not support direction or geometry in the way that the GPS space does.

To simplify implementation, rather than use Djikstra or similar algorithms to determine the centre of the Bounding Box, our application simply takes the first Point added to the Box as its centre. When computing the distance between the Bounding Box and another Point, the Distance Metric computes the space between them as the shortest of the distances from the Point to each of the Points of the Box. As the GSM application implements the space as a Discrete Asymmetric space, the Asymmetric Distance metric returns a pair of scalar values from its `getDistance(…);` method implementation and the second scalar denotes the distance from the defined centre point in the Bounding Box to the user's location Point regardless of whether there is another vertex in the Box whose distance to the user is less than that of the first vertex. We do this being aware of the way in which the model and Context-Broker use distances between Bounding Boxes and Points - to determine if a Box is incident with a user's Aura, and thus the operative distance is that from the user to the Box not that measured from the Box to the user. By the model, if a user's presence is projected onto a BoundingBox in the GSM pace, her location is taken as the centre of the bounding box so this distance metric's approach, using the centre to calculate distance from a bounding box to a point, maintains the consistency of the model.

A GSM Point class has a single member, a string which is its unique ID in the space. The space's PointSet class implements the abstract class DiscretePointSet. It provides a method which given a Point checks whether its ID belongs to the set of vertex names for the graph, one which given a Point and a Scalar value will return the set of points that are within that radius of the Point, and one which given two sets of Points will return the cardinality of their intersection. As the space is structured it implements the `isStructured();`

method to return true, but being unweighted it implemented the `isWeighted();` method to return false.

Payload objects are implemented in a common format by both applications. These are TextClue, PhotoClue, VideoClue, Note, and Bomb. TextClue customises itself by creating a new object in which the text String clue has had every N letters removed where N is determined by the formula

$$N = \frac{(100 - AQl_T)}{L}$$

where L is the length of the original string and AQl_T is total quantitative awareness the user has of the Clue.

PhotoClue, contains the URLs to four versions of a Photograph, the original and three others progressively blurred until their contents is unrecognisable, it also contains an integer member that identifies which URL to return when its value is accessed. A PhotoClue object customises itself by creating a copy of itself whose integer member selects the unaltered photograph's URL if the awareness type supplied is high, and the URLs of progressively more blurred photos if the awareness type supplied is Active, Passive or Low respectively. VideoClue operates similarly to PhotoClue except that its URLs point to differently cryptic video clues. User notes contain a single text String member which is either a sentence, or a URL that points to a photo or video. Their implementation of the `getCustomPayload(…);` method always returns a faithful copy of themselves including the value of the text String.

Bombs are implemented as a class with a single integer member that determines the impact the bomb's detonation will have on a user's life energy. Bomb objects customise themselves given a particular awareness type and level by setting their impact value to the value of the awareness level supplied to them.

How the Applications interface between the Context-Broker Service and the User

Each server establishes their gamespace, GPS or GSM, by using the tuplespace API to publish first the space's structure defining classes and then the current states of the objects and users that inhabit the space. These will be received by the Context-Broker and processed as described in the previous chapter. When a user performs a search on her mobile device the server receives this as a web request containing her user ID, location reading, chosen Focus size for the search, and the time that location was sampled. Her location is packaged into a Point object, her Focus size into a Focus object and these are submitted to the tuplespace API's update. User method along with her ID and location sampling timestamp. When a user initially joins the game, the server registers with the API to receive new and updated tuples from the Context-Broker matching the pattern of custom payloads labelled as intended for that user. The change to the user's state will, at the Context-Broker's end, trigger a reassessment of her presence and awareness. This results in the server first receiving a Flag notification that indicates the user’s awareness is being calculated and is currently in flux. The server may then receive updates to current payload tuples where the user's awareness of objects has changed; deletions of existing payload tuples where the user has lost awareness of an object, or new payload tuples, where objects have moved into the user's awareness. Each of these Payload objects is retained as part of the server's record of the user's state, and when the server finally receives an updated Flag notification indicating that the Context-Broker's calculation of the user's model-wide awareness is complete, it extracts the relevant data- URLs, text strings, and bomb details- from each customised Payload and packages these as a response to the Web request made by the mobile client.

When a user creates a note, her mobile client performs a web request which uploads the content of the note to the server in addition to its location, given Nimbus size, and a timestamp indicating the time it was created. If the note is a photo or video, the binary file itself is uploaded to the server and made publicly accessible through its URL. The note is assigned a new and unique ID as a new

Producer in the gamespace. The server packages the URL or the text of the note into a Note Payload object which extends the API's Payload class and publishes this through the API as a state update for a new Producer containing the location, packaged as a Point, the Nimbus value packaged as a Nimbus object and the note's ID and timestamp. The publication of this new object will trigger the recalculating in the model of all user awarenesses, which means the server will now receive from the API new Flag notifications for all of its users, and possibly new or updated awareness notifications. Because the structure of the mobile application-server communication is request-response rather than push-based, however, the user will not see these new objects until she makes a new search request.

New Bombs and Clues are created in a similar manner to user Notes as Bomb and Clue objects which also extend the Payload class in the API, but this process occurs through a helper application which provides a web-based administration interface for each of the servers. The administration service also provides the facility to create user updates in either space with manually specified timestamps, so that gamesmasters are able to create projection points manually if they wish.

Since the Context-Broker acts only to determine the user's context, i.e. what she should be able to see in the game and what effect viewing it should have on her, it does not need to be supplied with information such as the value of a user's life energy or which objects she has marked as bombs or safe. That state information is part of the game's logic, but not the user's context and thus is handled exclusively by the server and client applications.

Using the applications

The web-based administration interface allowed us to place a selection of Bombs, Clues and user notes in each gamespace, and create a selection of projection points so that our user could see objects from both spaces. The game was enacted over a residential area and our user was provided with a phone running the GPS version of the mobile client. The user was observed using the application and video recorded. During gameplay we observed that the user

was experiencing some difficulties with the application and we discuss these now.

Results

Issue #1: Not understanding where things are in relation to yourself in the model

In Chapter 4 we highlighted the location-aware game Savannah in which the users had trouble understanding Mixed-Reality spaces of low context continuity. Such spaces are those in which context transitions abruptly from one state to another. Players of the game traverse a physical gamespace as lions looking for virtual prey. Prey animals are assigned a Nimbus and are only visible to players when the player is within that Nimbus. The lack of cues signifying the edge of an animal's Nimbus meant that players would often be confused about the layout of the Mixed-Reality space. In one instance a group of players attempted to collaborate to attack an elephant but as they were stood at the edge of the elephant's Nimbus with some of them inside and some outside, some of them were unable to see the elephant and the game provided no indications as to why this was. Our model approached this issue by increasing the continuity of user context through the value of Awareness Level. With Awareness Level, a user's awareness of an object takes a value between 0 and 100, rather than being a binary property. We believed that we could naturally increase a user's understanding of their location in relation to other entities in an augmented reality space by providing developers with the means to signify proximity in richer detail through the adjustment of payload values. We discovered in our application trial, however, that despite this increased continuity, the user still struggled to understand her position in the system in relation to the objects she could see.

Instead of providing a map which shows streets, our GPS application illustrates only the relative distance and direction between the user and the objects she has awareness of. This choice was made to place the game's emphasis on exploring the gamespace via the model rather than through the user's existing knowledge of the physical space. This approach was designed to increase the

number of times the user contacts the server in order to increase the information available to the context-broker. More frequent updates to users' locations are particularly important for supporting our Context-Broker in the creation of projection point pairings between spaces.

As part of the basic gameplay for our Bombsquad game, we expected the user to act as follows:

1. perform search
2. from search results, choose virtual object to move towards
3. make an educated guess based on user notes and clues as to which direction object lies in
4. move in that direction
5. perform second search
6. given the user's direction of movement and the new search results determine if moving in the right direction
7. adjust direction of movement appropriately

Instead of observing the user behaving as we expected, we found that she struggled to make enough sense of the displayed search results to decide which object she would initially pursue. The user decided to move around the gamespace regardless of this confusion and was further confused when objects of smaller nimbus which had not been visible during her first search came into view after a second search. The appearance of these new objects disoriented even further her tenuous grasp on the geography of the augmented gamespace. The user did not attempt to alter her focus size during this initial gameplay perhaps because she did not want to further confuse herself by introducing another variable.

Issue #2 Finding navigation difficult, vs. map-based navigation

As well as difficulty understanding her location in the Mixed-Reality space created by our demonstration application, the user experienced confusion about the space that arose when she moved around it. Earlier in this chapter we discussed that context is normally used in 4 ways, for Discovery, Exploration, Artefact Creation, and Automation. We can also classify applications along

alternate lines based on the type and level of interaction they have with the user. The form and usage of context-aware applications can be divided into three different categories which we refer to as *notification*, *gathering* and *context-navigation-driven*. Notification applications are those which operate largely in the background, interrupting user activity with alerts or executing specific behaviours when certain context conditions are met. An example of this is Tasker. Gathering applications are generally used episodically and statically, that is a user with a particular context employs the application to perform a single operation of gathering information relevant to that context and then either employs this information in a manner that is not dependent on further context-sampling or dismisses the application. A vouchercloud user who employs the application to discover which nearby restaurants are offering lunchtime discounts performs the gathering and dismissal pattern of usage, whereas a Highlight user who employs the application to discover and chat with people nearby who share their interests performs the further non-context-dependent usage scenario. Context-navigation-driven applications employ user context continuously throughout the time they are foregrounded on the user's device. For example, an augmented reality system such as Cityview constantly samples user location and orientation in order to determine what virtual objects the user should be able to see. Since our Bombsquad application uses a request response communication archetype, notification behaviour is not a part of its interaction style and may be disregarded, but our description of expected user activity in the previous section includes both gathering and navigation activities.

Other applications or usage scenarios straddle these lines, for example the Google Cards app learns to predict user behaviour based on the patterns it observes in gathering-activity. It then presents what it believes to be relevant information alerts at the appropriate times, changing its behaviour from gathering to notification.

Our model calculates user awareness from the manner in which, and the extent to which, entity auras overlap. Where a user's Focus overlaps with an Object's Nimbus, the user will receive a customised payload from the object. In our trial

this is how the user sees virtual objects such as clue and mines. Where an object's Nimbus does not overlap with the user's Focus the user does not receive any payload from the object, and in our trial this equates to the object being invisible to the user.

The user confusion that arose in the course of moving around the space was particularly triggered when objects with smaller nimbuses were revealed to her. For example:

1. The user performs a search and sees one object in her results.
2. The user moves towards this object
3. The user performs a second search
4. The user can now see two object in her results, the original object and a new object, but the new object is closer to the user than the original object
5. The user assumes that the new object is the original object since it is the closest to her, or that she has moved in a different direction to that which she thought she had.

Where awareness exists between a user and an object, our context-broker provides the payload of the object and the object's location in the user's native space, **if available**. However, our trial demonstrates that the way this is implemented to hide objects from the user until she has awareness of them does not create an understandable picture of the model space for the user nor support the developer appropriately in creating one for her.

Issue #3

Seams

Seams arise both through simplifying the capabilities of different technologies in order to enable them to work together and also via gaps between the expectations for a technology's performance under ideal circumstances and how it performs in real world situations. Whereas seams that arise from the simplification of a technology's capabilities do so between the representation and the model, those arising from gaps between expectation and reality lead to a divergence between the sensed world and our representation of it. Some

classes of seams are: *Jitter* or *Dynamic Error*, *Static Error*, *Granularity*, *Loss of Sensing Signal*, *Dropped Transmission* and *Expensive Resources*.

Errors manifest as a contradiction between the user's state in the representation and either her experienced state or her sensed state in a different representation. For example, when a user is standing facing a landmark, but the gyroscope feature of her phone detects her as facing away from it, thus preventing her context-aware tour guide application from displaying information about the landmark. Jitter Errors, are frequently and seen in GPS sensing; they cause the user's state in the representation to appear to change over time where her experienced state does not. Static Errors are those which emerge in a single sensor reading rather than a series over time. Granularity Seams are those in which the user's state in the representation does not represent her experienced state because the representation is not fine-grained enough to do so. This may be a temporary issue which ranges over a variety of locations, for example changes in GSM cell-size or a decrease in the number of visible GPS satellites due to bad weather, or a more permanent one at a specific limited location, such as a decrease in visible GPS satellites caused by the proximity of tall buildings. Loss of Sensing Signal leads to a situation in which the user has a state in the representation which may be null, outside the accepted states for that representation or does not change with changes to the user's experienced state. Dropped Transmission to a client or server creates a seam in which the user's interaction with the application is limited or halted. And finally, Expensive Resources create bottlenecks at communication, processing or storage points in the application which lead to a decoupling of the user's experienced state from her state in the representation.

Some of these seams arise between physical world and representation and to support developers in managing them, our Model brought forward the differences between the representation created by sensing infrastructures and the model of the representation which abstracts features of the representation. Additionally, our Context-Broker allowed developers the freedom to implement the internals of their context representation's spatial structures however they wished and also allowed them to remain responsible for

receiving and processing the user's state into a presence in these spaces. We believed that this arrangement would provide developers with the opportunity to engage with and manage these seams through error correction. Our model also lessens the effect of errors and granularity seams by increasing the continuity of awareness so small deviations in a user's state due to these seams do not lead to large jumps in Awareness. Finally, by taking on the burden of processing and storing system state and reducing the amount of resources that must be dedicated to communicating changes to that state compared to in a distributed system, we hoped to reduce some of the bottlenecking effects caused by Expensive Resources seams. Our user's experience, however, demonstrated that seams can operate in more complex ways than we had anticipated and require a much more considered approach to their analysis if a platform and model are to support developers in identifying and engaging with them.

A New Seam: Moving During a Search

As our user continued through the game, she began to perform searches whilst she was walking to a location in an attempt to clarify her navigation through the gamespace. When a search is performed the user's location is sampled and passed with her Focus as an HTTP request to the gameserver which packages it into a user update and inserts it into the dataspace for the context-broker to use to update her awareness and therefore the set of objects she can see. These updates are then packaged by the server into a form that can be displayed in the mobile client application. They returned to client application as a response to the client's original HTTP request. On a PC with a good internet connection, this process is almost instantaneous, however, over a 3G connection with spotty network coverage, the delay between submitting a search and receiving the results is appreciable. This delay meant that by the time the user received her search results she had travelled such a distance from her location that the results were no longer relevant and in fact in some cases were very confusing to her. Dix and Abowd discuss a similar class of situation in a desktop environment where a user’s input to the system runs ahead of the system’s ability to respond to it, creating a what they call a violation of the mapping from the status of the user’s input (e.g. Having typed a full sentence) to the

status of the system's output (e.g. So far only displaying half of that sentence on screen as changes are detected and rendered slowly) [DIX and ABOWD'96]

This delay in reacting to changes in her state, which we shall call an Expensive Resources seam, causes the user confusion because the system's behaviour appears to her either incongruous with her situation or, and perhaps worse, the incongruity is not appreciated because it is not immediately apparent and the user is led to act upon bad information which only later in play causes her confusion and the realisation that there is something amiss. Since the client's display of search results showed only the user's location when she performed the search, the user had little indication that her location was so significantly different to that of the search results, this led to the delayed realisation that she was making decisions about moving through the gamespace based on out of date information. She then had to decide to either remain stationary during searches, which was frustrating and slowed gameplay, or keep moving and attempt to reconcile the out of date results with her current location by herself.

Conclusion

In previous chapters we presented a system of awareness that supports reasoning both within and across different representations of context by encoding those representations as spaces defined by feature sets such as discrete vs. continuous point sets and symmetric vs asymmetric distance metrics. We explained how entities can gain presence in representation spaces that they do not have the sensor technology to inhabit ordinarily through the use of projection points and we have detailed the mathematical procedures that our model employs in order to perform context and awareness reasoning within spaces, to aggregate that awareness across spaces. We also described the implementation of the model as a Context-Broker service and an API which allows developers to take advantage of the service. In this chapter we described a game we call Bombsquad designed to demonstrate this Context Broker and we gave the structure of the set of server, client and helper applications which we created to implement the game. We encountered three different problems during the demonstration of the game stemming from possible issues with the

model, the SPACES platform, the Bombsquad game's design and from Seams created in the ubiquitous computing environment.

In the next chapter we discuss how Pervasive Mixed-Reality systems exist as complex intersections of multiple milieu—virtual, physical, social, historic, infrastructural, and cognitive amongst them—which are, according to Chalmers and Crabtree et al [CRABTREE and RODDEN'08] , assembled by users through a process of cognitive interweaving into "fragmented ecologies". Our user's issues with the Bombsquad game involving her understanding of the model, the physical environment and of their interweaving as well as limitations of infrastructure and their relationship to user behaviour and the model, are a good illustration of this complexity and how it can cause platforms and models to fall short in their attempts to support developers and users. Responding to this complexity, we explore a philosophical approach to Pervasive Mixed-Reality systems that allows us to analyse the issues we came across more deeply and present their solutions. This approach offers us a new way to think about Pervasive Computing, about context-aware applications and about our model's relationship to these intersections and seams.

Chapter 7

Understanding and Responding to User Issues

Introduction

The preceding work aims to create a platform that supports developers of context-aware applications in working with heterogeneous, dynamic and partially known elements of context, and further, to explore the issues and design requirements of these in a seamful manner. This goal is approached through the design of a model for representing and reasoning about such context elements using a system of presence and awareness within abstracted spaces. The model then provides the basis for the construction of a context-broker reasoning service and the specification of an API that allows developers to package and deliver sensed data and media payloads to the broker for reasoning and manipulation. The SPACES model and platform manage the heterogeneous qualities of many context elements through procedures for discovering or specifying when points from two separate spaces should be considered to occupy the same point in some aggregate space. The system then allows for users to gain awareness of objects in additional spaces by projecting their presence across these points.

In the previous chapter we detailed a demonstration of the platform which involved a location-aware game developed for the purpose. The demonstration highlights three key user issues with our model which we identified as: Difficulty understanding her location in the model and relating it to the physical space, difficulty navigating with the model and difficulties arising from the user performing searches whilst walking. The first of these -- difficulty reading the model -- stemmed from the user's struggle to connect elements of the model with physical elements in her environment. On the other hand, difficulty navigating the model we feel arose from the use of negotiated

awareness in entity visibility. Finally, searching whilst walking characterises a seam in the application's space which is neither presented for exploitation, nor hidden or managed by the platform and application.

In this chapter, we discuss these issues and propose changes to the model and the platform that may resolve them. To address our user's difficulty in understanding and navigating the model as it was presented to her, we present various conceptual approaches to pervasive computing inspired by the fields of philosophy and architecture. These approaches tend to focus more on how spaces are understood and how they can be usefully presented and appropriated for inhabitation and we believe that this methodology dovetails neatly with the phenomenological findings of various Pervasive Computing researchers with respect to seams and Mixed-Reality systems. An examination of our model's design choices in light of this analysis and our commitment to design which is mindful of seams drives our changes to the model and our proposed changes to the platform.

Understanding Mixed-Reality Spaces

The SPACES model approaches context reasoning with two primary assertions. First, that proximity to other people, objects, or parts of the environment is, or can be cast as, the defining aspect of many user contexts. Second, that is it possible to spatialize even non-spatial elements of context in order to reason about proximity within them in this way. To perform this reasoning the model adopts a paradigm of user presence and Awareness drawn from the work of Benford et al., Rodden, and Dourish. User Awareness, derived from presence, provides a way to both qualify the nature and quantify the magnitude of proximity between entities under a given scale and is negotiated through the entities' Aura. Continuing from this, we posited that a pair of structures—a state space and a distance metric upon it—identifying the type of underlying set and defining the distances between members of the set are the only information about the space necessary to perform this sort of reasoning. This allows us to create a system that is able to reason about a wide variety of contexts without needing to understand the details of each context element.

A trial of the platform discussed in the previous chapter, however, demonstrated that a user experienced difficulty, both with understanding the model space as it was presented to her and with navigating the space on occasions when objects of small nimbus were revealed to her only as she moved closer to their location. These issues may well be compounded by the fact that the user encountered a seam whilst she was engaged with the app, which disrupted her use of it, but they also raise the question of whether the paradigms of spatialized context and user Awareness are unsuitable for the purpose of managing context or whether it is the application's presentation of these paradigms that should be adjusted to alleviate the user's confusion.

The work of Renz, demonstrates the ease with which topologically structured spatial configurations are understood and the importance subjects assign to this aspect of relationships when classifying spatial configurations. This indicates that the issues are not in the paradigm itself, but in its presentation, as does the successful employment of the paradigm by Benford et al. in the MASSIVE system. Renz's work and that of the Benford et al., however, do not present the user with a Mixed-Reality experience of the same nature that our user was presented with in the Bombsquad game so it is difficult to directly compare their presentations of such models. In order to identify the source of our user's difficulty positioning herself in this Mixed-Reality space, we turn to the fields of philosophy and architecture wherein much work has been undertaken to analyse our complex relationship with spaces, natural, built, technological, social, cognitive and negotiated.

Thinking About Space

What we talk about when we talk about meaning

In chapter 3 we spoke about a single location in the real world being *signified* in many different sensor representations by different encodings. For example, the same temperature can be signified by different numbers representing its value in Celsius and Fahrenheit. More generally, Saussure defines a signifier as a structure or symbol which denotes and connotes some underlying meaning.

Signifiers are integral to language as we cannot speak or write, or indeed think a thing; we must instead employ sounds, glyphs, or images, which can be communicated, stored and copied, in order to indicate the concept of the thing. The most famous example drawing attention to this being René Magritte's painting of a tobacco pipe entitled: *Ceci n'est pas une pipe.*

Every signifier we employ must signify some underlying concept. Without it, sound and image are simply noise, and marks on a page only scribble. But how are we to develop those signifier/signified pairings? In particular: how might a system for reasoning about context create such pairings at all and create them in a way that presents its users with a cohesive and comprehensible environment in which to act. This is a question at the core of our analysis of the user's issues with the SPACES system.

Philosopher Gilbert Simondon's theory of signified concepts holds that experiences matching a signifier or group of signifiers: "dog" perhaps, coalesce together and evolve as we might imagine a tree grows [CHABOT, KREFETZ et al.'13] p(102-103). The initial experience of a the signified and signifier creates a singularity of experience which future experiences of the pairing may graft themselves onto, strengthening the initial signifier connection, like the thickening of a tree's trunk as it grows. Where they contradict the initial experience, Simondon holds they do not graft onto the metaphorical tree's trunk but diverge from it as a branch's growth diverges from the direction of the trunk that carries it. In this way the term "dog" can come to refer to and potentially conjure both our childhood neighbour's miniature poodle and the imposing Alsatians of modern policing. A hauntingly beautiful visualisation of this concept can be found in Idris Khan's *Every…* projects, in particular we present here the piece *Every…Bernd And Hilla Becher Spherical Type Gasholders* (available to view at [KHAN'04]), which layers a series of images from Bernd and Hilla Becher's photography series *Spherical type Gasholders* (selection available to view at [BECHER and BECHER'72-']) on top of one another so that their shared structures are picked out, reinforced. The process of reinforcement creates an image of a fuzzy visual archetype, whilst in the places

the images vary, ghostly stanchions, girders, and piping sprout from the main, solid structure, hinting at possible differences contained within the form.

Simondon referred to the flexible nature of those signifier/signified structures as *metastable,* borrowing the word from physics where it means a system which has reached a point of temporary stability other than is "absolutely stale" point of least energy. The term is applied here in philosophy to indicate that the branching process permitted in the integration of a new signifier/signified pairing allows these metaphorical trees of signification to absorb potentially disruptive experiences– those which differ markedly from, even contradict, the established archetype -- for example hearing an untrustworthy person referred to as a dog, without catastrophic damage to or collapse of the main trunk of the signifier's meaning. He also intends it to indicate that the trunk of these signification trees can be subject to change given enough input of a particular kind. Imagine your first experience of a "dog" was your parents' ironically named pet cat. As primary as this experience may be, the volume and weight of contradictory experiences of the meaning of dog would soon overwrite it; the initial example becoming a branch from the main trunk. Metastability does not only reach backwards, gathering previous experience, it reaches forwards in the form of anticipation. When we hear "dog" we form an expectation of the experience to accompany the signifier. To be presented with something which entirely fails that expectation can provoke humour, surrealism or confusion. When our user experienced confusion during the application trial it was a signal that the system in some way both failed the expectations created by existing signification structures she encountered and was unable to support her in creating new structures during its execution.

Another aspect of the flexibility of signification is captured by Ihde in the concept of *multistability* [IHDE'86] . A multistable signifier is one which denotes to two or more unconnected signifieds to a reader[3] These signifieds are

[3] We use reader here in the general sense as one who apprehends a certain stimulus and doing so, translates it into a specific meaning, so that not only text

not simultaneously accessible, instead the reader selects one or another consciously or unconsciously. A classic example of this can be seen in Rubin's Vase [Figure 36363226] which can be read as either a vase or two faces. Arguably, Pervasive infrastructure enacts a form of multistability upon the physical environment too, bringing new meanings to the spaces it augments, for example in the way that a location-aware Mixed-Reality game can turn a playing field into a virtual savannah. Chalmers' theories of Mixed-Reality user experience [CHALMERS and GALANI'04] hold that the mixed space is constructed by interweaving multiple elements: the physical space, the virtual world and other elements such as the user's memory of the space. This sounds very much like a process of generating multistability. In the case of our trial it would appear that the user struggled to generate a new multistability that combined the world of the model and the physical world through which she was situated.

Simondon's metastability charts a course between Saussure's Structuralism and Derrida's Deconstructivist philosophy. The Deconstructivist criticism of Structuralism is that it assumes there exists a stable and unchanging meaning behind every signifier, which patiently waits to be uncovered by semiotics and critical reading; that there is a unique and correct reading for every text (film, photograph, building, application) which it is a reader's job is to identify [DELEUZE'53] (p171). Instead, Derrida holds that there is no passive meaning behind any signifier and what exists there instead is a chain of yet more signifiers. Even the simplest of signs, he asserts, are only understood through an appeal to their contrast with the things they are not. This, he termed *différance*: a type of meaning that is driven by difference between experiences and indeed the only sort of root to meaning that Derrida's philosophy admits [DERRIDA'82] (p1-28). But if there is no foundation, no stability to meaning, the developer assessing ways how to manage and support a user's understanding of a system is lost before she can even start. Simondon's

---

can be read, but a multitude of other stimuli including but not limited to films, spaces, behaviour, and pieces of music.

metastability of signifiers instead offers a signified that is graspable, identifiable, yet still dynamic, flexible, with no requirement for the transcendence of experience that Structuralism is charged with enforcing. Metastability is what gives us a basis for communication, even where shared experience does not exist. We each understand "dog" in a way that, though it may differ slightly from our neighbour, is still similar enough and flexible enough to align with their own understanding when together we speak of dogs.

Figure 36 - Rubin's Vase (public domain [SMITHSON'07]) can be interpreted as either a vase or two faces

Metastability and Technology

The previous section refers to a metastability that is cognitive and individual but it also points to a metastability of signifier-signified relationships across populations and this form of metastability exists, we assert, not as a binary—metastable or not—but a continuum. The connection of a signifier to a particular type of experience may be strong or weak across a population. The replacing of a handle on a door with a metal panel, for example, is very commonly accepted to signify that it may be opened by pushing and this meaning persists strongly across many decades and cultures. We can see examples of trends towards lower metastability of particular signifiers in contemporary culture in the video *A magazine is an iPad that does not work* [EFMD'11] , which shows a toddler attempting to interact with the pages of a

paper magazine by swiping and pinching its images. Metastability across populations enables communication and understanding, but what drives metastability? What is at the root of comprehension of Pervasive, and indeed all, applications?

Simondon's interests coalesced around the history of technology or "technical elements" a term which he identified through the quality of technicity which MacKenzie translates as: "The capacity of an element to produce or undergo an effect in a determined fashion." [MACKENZIE'02] p13] Thus, the most primitive of tools like stone axes, through machines like the loom and steam engine, to the interweavedly situated and distributed Pervasive systems that we discuss in this work are all technologies to Simondon's mind. This definition of technicity sounds remarkably like that of metastability which we could consider to be the capacity of a signifier to evoke a particular form of experience in a determined fashion. Technicity is to the behaviour of a technology as metastability is to the behaviour of a signifier, although as in HCI, oftentimes the behaviour of a technology involves and depends upon the behaviour of its signifiers.

The two processes which produce technicity, Simondon identifies as t*emporal and spatial reticulation* [SIMONDON'05] (p31) in which an object is repeated and reproduced across time and across space. Retaining its capability to produce certain effects across these repetitions is what defines its technicity. Since technicity enfolds metastability, however, this repetition is not simply a spreading outwards of the object or of its signifiers. Every individual spherical gasholder repeats, re-enacts, the spherical gasholder archetype at a different time, in a different place, but each re-enactment not only strengthens the metastability of the form, it is recognised as a repetition of the form precisely *because* of that metastability. We would argue it is repetition, which performs temporal reticulation and communication (or more generally dispersal), which performs spatial reticulation, that together act as a lure to metastability and in turn that it is the very existence of that metastability which lures us on to a sign's communication and repetition.

Routes to Metastability in SPACES

A developer seeking the legibility that metastable signification promises, then, is looking to leverage the repetition and communication of an application's signs. A developer of a system whose users struggle to read its structures must seek to analyse where it fails with respect to those criteria. But how is this achieved and what aspects of Pervasive Systems can be identified as affecting it?

McCullough [McCullough'04] argues that situated and mobile systems do not benefit from the relatively blank slate afforded by cyberspace; that they must instead inscribe themselves upon a built and natural environment that is already rich in structure and meaning of many types, amongst them social, economic and historical. We would go further than this and say that non-situated applications nowadays must still inscribe upon the structures of meaning that have developed since cyberspace's inception, but that what complicates Mixed-Reality and Pervasive systems is that they must inscribe on both the physical and the virtual at once and this process must generate its own meaningful structures at the intersection of these two milieu.

McCullough is not alone in identifying these sorts of issues as ones which arise when thinking about Pervasive Systems. MacKenzie, extending Simondon's work with technology, draws a distinction of complexity between Pervasive Systems and those technologies whose technicity may be deterritorialised -- made placeless. An axe, for MacKenzie, represents a conjunction of processes and qualities - a hardness of steel, a particular grade of hardwood, a particular angle to the blade -- which can be transported to new locales with relative ease, either in the conjunction of that particular axe itself, or by repeating those same processes elsewhere to construct a new one. Pervasive systems on the other hand are possessed of a situated form of technicity that cannot be separated from its locale as easily as a simple hand tool or even a more complex machine, and is thus, MacKenzie holds, difficult to analyse exhaustively except in situ [MacKenzie'02] (p14). To analyse these situated applications, then, it may be helpful to look at the ways in which the physical structures they inscribe upon, those of the built environment, are thought and analysed

themselves. Architecture, after all, is its own language and possesses in its grammar its own capacity to "produce or undergo effects in a determined manner" just as technical elements do. Indeed in [SHEPARD'11] Shepard classifies Context as a framing mechanism that analogues architecture in that it, like the built environment, is an indicator of how we (or a context-aware system) should act.

Framing, Interaction and Repetition

Historically, what buildings were assumed to frame was space itself and we might draw a parallel between this and our model whose point-sets, distance metrics and projection points, define and delineate the spaces (virtual and sensed) around users and objects. Shepard [SHEPARD'11] (p16-37) credits bodies such as the British avant-garde collective Archigram [SADLER'05] with spurring a seachange in architecture's treatment of space, particularly in urban environments, from that of a medium delineated or demarcated by built objects to one defined through what he describes as "the less determinate, more ephemeral, untidy ebbs and flows of urban life."[SHEPARD'11] p18 This approach, he notes, was supported and expanded on by various contemporaries of Archigram, including Chalk, who he claims places transient objects and the movements of people as primary in space [ibid] and later Lefebvre who describes space as a social product [LEFEBVRE'91] , implying that what architecture frames is not space itself but the interactions and actions within it. In Computer Science this thread is teased out by Dourish in his previously discussed assertion that context itself is constructed in interaction, in the relationship of one user to another.

Gärdenfors, too, emphasises the importance of relationships and their framing. In The Geometry of Thought [GÄRDENFORS'04] , which like Renz's work concerns itself with the cognitive adequacy of systems of representation, he asserts that association -- the relating of one thing to another -- is the most important aspect of cognition, and that rather than isomorphism between entities and their symbolic representation, it is isomorphism of the relationships between those entities and the symbols representing those relationships that a cognitively adequate model must preserve. For example, how a system

represents a room in an office building might bear little resemblance to the room itself, but it should be conceptually closer in the model to the representation of other rooms in the building than to that of the weather that morning. This supremacy of association in some ways also calls to Derrida's différance, that meaning is preserved not in objects themselves but in their relationships.

Cunliffe's work on Social Poetics [CUNLIFFE'02] (the socially derived imagery we rely on to communicate every day experiences) tells a similar story. Examining the role of symbol and narrative in workplace communication, Cunliffe concludes that narrative ecologies sharing the same symbolic framework evolve amongst colleagues even where the specific symbols used are not shared. For example, a team-wide framing of work processes in the language of emergency responders - firefighting, triage, crowd-control -- despite the fact that each colleague uses different individual metaphors. Although as Weiser states: "there is immense value in allowing things to be themselves" [WEISER'94] , people appear to gravitate in particular to frameworks that allow relationships between things to be themselves, and Cullen's work shows those frameworks where they do evolve naturally tend to evolve, as metastability would seem to predict, along paths that repeat and maintain their relationships' cohesion.

The reciprocal relationship between framing and interaction that we see arising here -- context frames interaction, interaction generates context -- echoes the relationship of metastability to reticulation. Indeed, the mechanisms of placefulness as described by Dourish involve the repetition of activity or interaction within the frame of a particular space. Place is space made symbol, given meaning "*by its patterns of recurrence in human use*" [CHALMERS'04] . The space becomes a Context, a frame, for the activity and the repeated activity strengthens the space's identity as a Place where that activity occurs.

Our system is not solely space, however. It is as previously stated, an intersection of various milieus including the built environment and technical objects like mobile devices. When our user experiences trouble reading the model, perhaps it is that the model constructs the wrong intersectional framing

of the user's interactions both with the space itself and the entities within it. By the wrong framing, we mean one whose structures do not readily lend themselves to reproduction across space and time. The model was designed to make calculations about context particularly simple for a context-aware reasoning system and to minimise the amount of information any such system requires from the applications administering the spaces it reasons across. The relationships to and interaction with other entities are all encoded in the model in this minimalistic way. But this information, though it may be enough for the system itself to reason as it needs to about context, may not be sufficient to support the user's recognition and repetition of those structures of relationship and interaction. Perhaps then what is needed in our system to prevent user confusion is a way of presenting these relationships, a framing for them, that is more cohesive with how they are experienced in the Mixed-Reality space.

Figure 37 - The user attempts to find correspondence between the model and the physical space by rotating the phone screen on its side and even upside down

Framing Failures in SPACES and BombSquad

Chalmers reading of the intersection of milieu that Pervasive and Mixed-Reality applications perform is that it operates through the user as a cognitive interweaving of various mediums: physical, virtual, social etc. We believe that when our user attempts to perform this interweaving what she is enacting is in

fact a form of dispersal and repetition. Slices of the virtual world are brought into her cognitive model of the physical, projected out from her location over the environment around her. Likewise, onscreen she brings slices of the physical world to bear on the virtual. In the trial video, we see her rotating the phone's screen in an attempt to find correspondence between landmarks on the screen and landmarks as she knows them in the physical space [Figure 37373327]. These are repeated pairings of ensemble structure to meaning: That virtual object is on that street corner, this coffee shop is represented by that orange marker. The process is also inherently temporal, it involves previous experience and projected expectation: I just came from that clue on the bottom right of the screen, so I need to go this way to get closer to the next clue.

The application, however, which can be said to act as a framing for the model in many respects, fails to support this activity in a variety of ways. First: the work which the application does to draw the physical world into the search results representation is extremely limited. In the GPS version of the game, the user is presented with results laid out spatially on a plain background as if she were standing in a field. As explained in Chapter 6 we felt this would encourage her to focus more on exploring the model by performing many searches—a desirable behaviour for our application and platform as searches from dual-space users assist in the creation of projection points which increase the information the context-broker has about the relationship between pairs of different spaces and therefore the richness of its spatial model. Of course, the user is not in a field, she is in an urban environment and it is up to her, the application decides, to translate the physical experience around her to one which can be lain over and interweaved with the features of the model. Instead of mapping capabilities, notes acting as landmarks are inserted into the space as points of virtual-physical reference to assist the user navigation, but clearly this framing which draws forward just the proximity features of the model and obscures both those of object aura and of the physical space so extensively is not user-friendly. A better narrative framing may have helped here, but undeniably, although the platform's job is supported by the abstraction of the model that hides the details of representations from it, it is more detail about

the space, not less that enables the user to identify and repeat pairings of ensemble structure to meaning.

Changing the application, model and platform to provide better *framing*

Since the information the user receives about any entity she has awareness of, and the information she has about her own presence, includes a GPS location coordinate, this framing issue can be solved at the application level by rendering the search results over a map sourced by the application from a third-party service. The hardware we originally worked with (Symbian S60 2nd and 3rd generation phones) offers limited mapping support, but new platforms such as android and iOS provide both onboard mapping services and the ability to share a user's ongoing location with a webpage so that the server-side portion of the Bombsquad application could render the results on a map embedded in a webpage that the user can access and interact with within the Bombsquad application on her phone.

To support older hardware such as the Symbian phones, the server application would have to retrieve an image of the area covered by the user's search results from some third-party service, resize it to fit appropriately on the user's screen and provide translation values for mapping the GPS coordinates of search objects and the user's realtime location onto the pixels of the image. For example: Let the geographic area *G* of search results to be displayed on the user's screen be normalised to the smallest square covering the set of search results given by an origin coordinate O=(2.0130, 60.382) and a distance representing the length of the square's sides. Let the area of screen to be filled by the results be a square *I* of side 300 pixels. The client application then follows a procedure of subtracting the GPS origin coordinate from the GPS location coordinate of each object, which gives an offset in degrees from the origin to the object's location. This offset is then scaled by the ratio of G to I and the resultant value gives the pixel coordinates for placing the object on the screen within the map image. Assuming the size of I is known and does not change, since the device's screen size is known or can be queried when the user starts a game, and does not change, the server only has to pass the origin

point and size of O to the mobile client and these numbers can be plugged into the same scaling equation to position all search results.

Our model, however, does not only support GPS spaces, but a variety of context representations. Similar approaches to GPS mapping could be used to display spaces which use other location technologies, or refer to other types of sensed data. For example: For a GSM space, details of the relevant subgraph of the GSM network can be passed to the mobile client for rendering, or the server application could provide an interactive rendering on a webpage that the mobile client can access.

One change that could be made to the SPACES Platform is to add the URI of such services to the awareness results tuples returned by the Context-Broker. This would guarantee that no matter what space the user gains presence within, the application will always be able to access a graphical representation of her presence and Awareness results in that space.

History as a Lure to Repetition

As we discussed previously, where metastability arises it is the result of an aggregating force, the enacting of a form of pattern-recognition upon, and with reference to, previous experiences. In order for this to occur, previous experiences must be available to us. The recording and making available of histories supports their repetition by allowing the user to see and review her experiences in the Mixed space. The history presents a temporal frame for the user's experience. But in the Bombsquad framing of the SPACES system, a user performing a search, updating her user presence, is only presented with results that give a snapshot of her presence and Awareness at the moment she performed the search—she has no history and without one, as we saw with the lack of detail about her space's effect on repetition, she may find temporal reticulation difficult to see, understand, and ultimately to perform.

It could be argued that this is the case in other applications that do not suffer from the same legibility issues -- a GPS mapping app for example. This sort of application often provides no indication of where a user has been, only their current position on the map. However, the continuous updating of that position

supports the user in cognitively maintaining her own limited history which consists of a dense interweaving of her experiences with her displayed position on the map. We might call this sort of history *intensive*, as being cognitively maintained, it is rich with the user's fully lived experience, but persists in this richness only for a short period. Conversely we can define *extensive* history as one maintained by the application; not so rich as the user's experience but transmissible, durable, and available not just to the user herself but other applications and users. This concept of extensive history is similar to Khalid's concept of extended episodic experience [KHALID and DIX'14] which encompasses the memories users preserve online through social media postings, however in our context the extensive history we refer to is collected by the application automatically from sensed data rather than specified by the user manually. In our Bombsquad application, those updates that might generate intensive history are both discontinuous and widely spaced. The game is played in an episodic manner that leads to long gaps during which time the user is not looking at the search results and her position in the modelspace but at the physical space around her - during walking for example. The lack of continuous feedback, the framing of the modelspace as something which comes to the user in short, sparse and static bursts, could reduce her ability and inclination to perform that interweaving of milieu so essential to the success of a Mixed-Reality application. If nothing else, it certainly reduces her opportunity to do so, and comparing the temporal with the spatial, it would appear from our user's experience that increasing spatial and contextual continuity alones through awareness does not represent a framing of the space that successfully lures repetition. Increasing continuity temporally as well as spatially may well give better results.

Again, this issue can be confronted at different levels of the system. In the application adding a routine to update the location of the user's avatar on the search results screen would provide that useful immediate feedback and increase her sense of continuity in the space. But on the subject of extensive history, the model and context-broker platform themselves are almost atemporal in that the only history they store is concerned with the creation of Projection Points. Indeed, interaction with projection points aside, an

application could insert a user's updated presence information as that of an entirely new user every time and receive almost exactly the same awareness results as it would if those updates were tied to the same ID, the only difference being that with a consistent user ID the API to the dataspace receives update rather than add notifications for those Producers that the user already has awareness of. In a system where interaction carries such importance, the memory of having interacted with an entity before should perhaps be brought forward. This can be done by storing a history for each user of the presence and awareness updates they have received in a structure that supports various inquiries about that history through the context-broker. This structure can store a timeline for each object the user encounters, recording the time she first encountered it; the set of presence values involved in the encounter – i.e. their locations and aura both native and projected; and the same values for each subsequent encounter or change of awareness. Since this information is common to all spaces these structures can maintained whether the user is located in a Continuous Symmetric space like the GPS space or a Discrete Asymmetric one like the GSM space.

Supporting History in the SPACES platform

When a user gains or changes her awareness of a Producer, the Context-Broker returns a results tuple containing the appropriate payload and current presence information of the Producer. With the preservation in the model of user history, these tuples can now be augmented with an object containing the timeline data concerning previous encounters the user has had with the Producer. We propose this object implement various helper methods such as `boolean seenBefore(ObjectID);` which returns true if the Context-Broker has record of the user previously gaining awareness of a particular object, `DateTime getLastSeen(ObjectID);` which given a previous encounter with an object will return the time of the last encounter, `AwarenessDetails[] getEncounters(ObjectID);` which returns the array of descriptions of the user's previous presence configurations and Awareness values with the object, where AwarenessDetails is a new class that wraps the timestamps indicating the start and end of the presence Configuration, the presence information—locations and auras—of the consumer and producer, and the Qualitative and

Quantitative Awareness calculated between them. This supports the application in providing indications to the user that she has previously encountered an object and how her awareness of it has evolved.

This idea to make the model more explicitly temporal and to drive more continuity between state changes as shown in the application, is also a useful way to address our second issue in which the user encountered confusion not just in understanding her location, but in her relationship to the space as she moved through it. The source of her confusion was the sudden appearance of objects with small nimbuses, which she mistook for objects (with larger nimbuses) that she had seen earlier. A framing that brings forward the objects' differences and histories, such as a visual translation between new and old results, would certainly assist the user in understanding the model here. For example, given the appropriate history information by the server, the application could provide an animation showing the icons from the previous search results animating into their new positions, followed by the sequential appearance of icons for new objects which are set apart from the old ones by glowing icons or similar visual cueing. This particular issue, however, also opens the way for us to think through some structures and assumptions of the model that were not otherwise apparent.

Visibility vs Availability

In our model, users do not receive payloads or any form of notification from objects unless they have some level of awareness of them. If there is no awareness, the user has no indication that the object exists. The situation the user experienced -- where objects appear to pop confusingly into existence – is created when objects with varying nimbus sizes are located within the same space. Although increasing qualitative awareness allows for objects to be revealed slowly, our app translates this into an obfuscation of the object's content rather than, for example, a blurring or changing the size of its icon. An object with a large Nimbus may cover the user's Focus and location whilst an object with a very small nimbus, although it is closer to the user, has no overlap with her Aura and will be invisible to her [Figure 38383428] until she

moves so that her Focus is incident upon its area of presence [Figure 39393529].

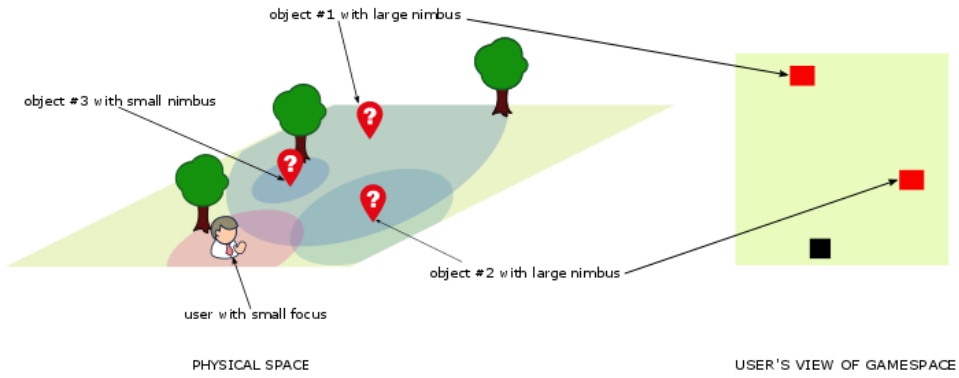

Figure 38 - The user has awareness of object #1 and #2 but not #3

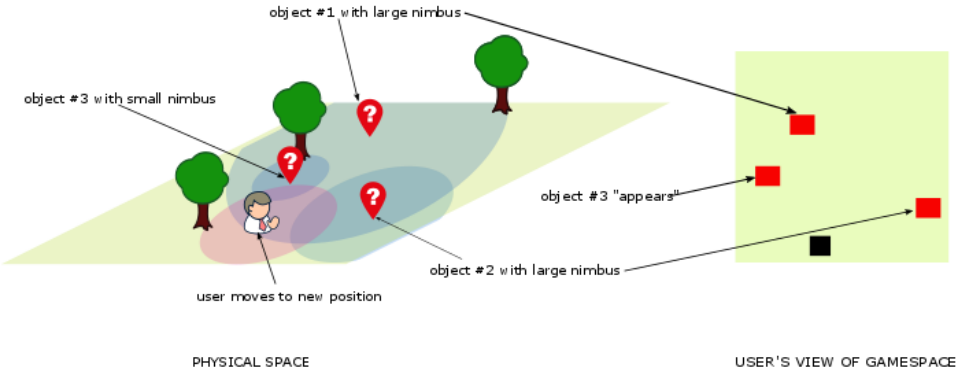

Figure 39 - Object #3 appears when the user moves closer to it

Recall presence and Awareness were employed in the model in part because they provided the system with a measure of efficiency, similar to that of the bounding polygons of computational geometry in calculating information about object collisions. Using the Awareness paradigm, the system should not need to calculate the fine details of the relationship between every entity in the space, only between those whose appropriate Aura intersect. That the hiding and revealing of objects in this way causes such confusion to the user, brings into question the validity of using presence and Awareness as a paradigm here.

Examining another situated application, Hitchers, some differences become apparent between the way Awareness is used in that game and the approach

taken by our SPACES model. The Hitchers application is situated and mobile but it also employs the awareness paradigm in a slightly different manner. When the user performs a search for hitchers she sees only the hitchers she has direct access to. Although Awareness mediates both the visibility of, and the user's interaction with, individual hitchers, there is no requirement for her to get closer or further away from the hitchers she sees in order to interact with them more fully or more safely. In contrast, in our Minesweeper game, the user's goal is to get as close to known clues as possible before interacting with them in order to see them with the highest level of clarity and stay as far from bombs, and unknown objects which could be bombs or clues, when interacting with them in order to minimise any potential damage. Thus, we have involved not only a gathering activity but an element of context-driven Mixed Space navigation in our game, which Hitchers does not employ to the same extent. Furthermore, the SPACES model positions the Awareness paradigm as a tool for performing this navigation through the act of mediating what the user can see in the mixed environment. Although this clearly has problematic consequences, it does not necessarily indicate that Awareness as a paradigm is problematic in applications that involve Mixed Space navigation, rather it is the relationship between framing and Awareness, as we shall see, that must once more be carefully managed.

Framing the Criteria of Awareness

With its focus on both reticulation and the intersection of milieu, Simondon's work draws out a tension between situatedness and distribution as a characteristic of technology. In much the same vein is Fuller's view of the city [FULLER'01] as one characterised by the gathering, enfolding and dispersing of action and interaction. This is a position informed by the work of Vilem Flusser [FLUSSER'99] who identifies the evolutionary transition to widespread tool-use, characterised by a dispersal of tools throughout communities following from the segregation -- the gathering and enfolding --of tool manufacture in designated areas. Just as this situational enfolding of the knowledge of tool manufacture helped the manufacturing process become repeatable and transmissible between ancient toolsmiths, so too Fuller argues,

does architecture gather and enfold action. Again this recalls Shepard's description of architecture and context as framings for action.

The aim of the pervasive system, then, should be to enact this same spatialized gathering and enfolding of action and interaction. In designing our model, we argued that proximity -- a user's nearness to a person, object or state -- is the primary constructor of Context, and that Awareness derived from presence is a measure of proximity over a negotiated scale. From Fuller's perspective, our model enacts Awareness as a gathering and enfolding of objects in proximity to the user, since the user will only receive payloads from objects of which she has Awareness. But that gathering is contingent upon not only the presence under the user's control, but the presences under control of other objects, and is enacted on an individual basis with each object. Even when considering Nimbus to represent visibility of an object, the Nimbus size may vary from object to object to represent that the object is harder or easier to see from various distances. From this context we can see that if the user confuses a new object possessed of a small nimbus with an old object possessing a larger nimbus, this is because she has failed to understand, or in our case the system has failed to correctly present, the criteria by which Awareness gathers and enfolds objects.

As stated, if awareness is a gathering of objects then it is also, from Shepard's point of view, a framing of the population of the space the user inhabits in terms of the Awareness function. Which is to say a framing of the space which is negotiated individually between the user and each object in that space via their aura. Unlike buildings which often act as pieces of visible infrastructure, by infrastructure we mean stable and reasonably static as per [BENJAMIN, YANG et al.'01] , and whose actions are arguably universal across its inhabitants, Awareness is dynamic, transient, and individualised. Moreover, in our Bombsquad application, unlike a building to its inhabitants, Awareness acts almost invisibly in the application, as if the user is surrounded by structures that control her experience and activities, some of which she can control and some of which she cannot. A reasonable response to this is to alter the Bombsquad application to return details of the Aura of the objects along with

their Payloads so that this information can be displayed for the user graphically on the results map.

Since Aura is maintained as a simple radius centred on the user's location, this is a simple change in a space whose distance metric is Euclidean. Each object displayed on the search results map can now be annotated with a circle showing the extent of its Nimbus. The user's focus, indeed can be set visually by allowing her to adjust the size of a Focus circle centred on her own location. In other spaces, however, this representation is not so simple and requires the platform which displays the Awareness results, whether that is the server or the mobile client, to understand how the distance metric permits aura to propagate across a space. In the case of a Continuous space whose distance metric is defined by travel time between points, for example, the application may have to resort to approximating Aura as convex hulls in order to respect their propagation under the space's Metric. In addition, some spaces may not be represented as images. In our implementation of the GSM space, rather than focus our efforts on rendering an image of the network subgraph, search results are displayed as a list ordered by the cells in which they appear. Moreover, one of our goals for the SPACES model and system was that applications should not have to understand the details of other spaces in order for their users to gain presence in them. To be presented with a set of search results that includes objects from another space which you must display visually to the user poses quite a challenge for the application that is has no information about that space. For this reason, we also propose a change to the SPACES platform API requiring that applications implement a RenderingService for each space they define. This service is connected to the DataspaceManager which we propose subscribes to tuples from the dataspace which comprise rendering requests for parts of spaces that the application manages and delivers these to the registered RenderingService.

We propose to introduce two new classes: ContinuousRendering and DiscreteRendering. As input the RenderingService receives a BoundingBox (defining an area of the space to be rendered) and an image size in pixels which represents the size that the output rendering should take. It then returns either a

ContinuousRendering object or a DiscreteRendering object, dependent on the type of space being rendered.

The ContinuousRendering consists of an image of the space and a BoundingBox which defines the area of the space that is covered by the image as well as a list of (object ID, point, point array) triples that give the rendered location of every object to be displayed in the sub-space and the set of points describing the path of the approximate convex hull of their Aura. A mobile application can use this information to directly locate objects from the sub-space at their appropriate positions rendered over the image.

Since a Discrete space is most likely to be rendered in 2D as a graph, the DiscreteRendering contains four sets of objects: One or more VertexRender objects, which are wrappings for a PointName, a centre and radius; zero or more LineRender objects, which are wrappings for two endpoints; zero or more LabelRender objects, which are wrappings for a string and a pixel location, and zero or more (object ID, PointName, PointName array) triples which give the vertices to render particular objects at and the vertices that their Aura cover. Given these, an application can construct a 2D vector image of the subgraph whose structure it understands well enough to render Objects that the user has Awareness of in the correct locations on the subgraph image.

Finally, to allow applications to request renderings from others in the dataspace, we propose that the DataspaceManager provide a request mechanism that takes a BoundingBox or a String array of Point encodings (built by calling each point's `toString();` method), which define the space to be rendered; a Resolution object, which wraps a pair of values specifying the size of the Rendering required; and an Object list containing (object ID, Aura value, String-encoded location) triples. To perform a request, the DataspaceManager publishes these structures in a tuple to the dataspace and listens for the tuple containing the correct rendering in response. Since BoundingBoxes are designed to be constructed by adding all the points they should cover to them, the Context-Broker can construct the covering BoundingBox for the user's Awareness results in her native spaces and supply this to the application in the dataspace flagging tuple for that user which

signals that an update process has finished and the user has a stable Awareness in the space. The same information for Awareness results from spaces that the user is projected into cannot be sent to the user as a BoundingBox because her application may not have a copy of that space's class definitions and the Symbian platforms we implemented on do not support classloading (although this is possible on newer android devices). Instead this can be obtained from the String encodings supplying the Producer presence information in the Awareness payload tuples. To allow the application to request from the dataspace a rendering of the user's position in spaces that she has been projected into, the Context-Broker should also supply, in any Awareness payload tuple which is the result of an Aggregated Awareness, the Aura and String encoding of the user's image presence that is involved in the proxy awareness calculation.

Awareness as Infrastructure

Framing, which is arguably an act of collection and presentation, is not the only architectural approach we can take to Pervasive environments. Benjamin et al [BENJAMIN, YANG et al.'01] , for example, describe the city as a system of flows and envelopes. In this paradigm, the envelopes, which are barriers such as walls, access points such as windows, and broadcast points such as LCD screens, may not only reveal but hide, transform, or transmit the "happenings", "passing presences" [CHALK'66] and "ephemeral, untidy [flows] of urban life." [SHEPARD'11] in the city. They illustrate these envelopes in a piece of installation art situated on the Hudson River which transforms and makes visible to New York's inhabitants the sensed behaviours of the river's fauna as a series of floating light strips. Similarly, Ito & Okabe [ITO, OKABE et al.'06] when speaking of mobile devices describe them not as portals which transport their users into other spaces, but as membranes which mediate the access of "absent others," a friend messaging them from the next town, for example, to the user and her immediate surroundings. Finally, Hillier's Convex and Axial spaces [HILLIER and HANSON'89] can be said to manage the flows of the city through the envelopes of access and navigation they create: the structure and signage of Axial spaces encouraging flow of strangers through the city, and in

contrast the slower-pace Convex spaces which are more available to residents and mediating access to more private spaces such as homes. All of these paradigms deal with movement of people and of vehicles or with the dispersal or communication of information, moreover the structures that enact them are either highly visible as in Benjamin et al's Hudson River installation, or take the form of stable infrastructures.

Awareness, since it mediates the access of objects to users, can also be thought of as an envelope. But as an envelope it exists only transiently, as an interaction between the presence of two users. This runs against Benjamin et al.'s view of envelopes as infrastructure, more stable and in some cases more visible than the flows they manage. As we previously discussed from the position of framing, that the nature of each individual presence interaction between Consumer and Producer is not made available by the platform, and therefore not presented to the user, is likewise at odds with these paradigms of flow and mediation. Furthermore, Benjamin's envelopes are as specific in type as they are universal in delivery. They mediate particular flows in particular manners but they do so democratically—every resident of New York can see the state of the Hudson's fauna by looking at the installation—whereas Awareness in the platform is applied universally to all mediation, regardless of its nature. That mediation is simultaneously a mediation of both individual visibility and transmission of information, service, and behaviours as custom payloads. It is perhaps understandable then that the user struggles with the paradigm, not only to understand where new objects come from and why they appear, but to navigate through the model in general.

In chapters 2 and 3 we spoke about Reeves taxonomy of performance and how we envisioned its relationship to the different qualitative levels of awareness - a configuration of Passive awareness exposes the Consumer to effects but not their originating actions, whereas an Active awareness shows the reveals the actions but not the effects. Reading Benjamin et al, this is clearly not the only way to position Awareness, nor is there only one sort of behaviour a user can undertake in a space using context. Recall our introduction to Hierarchical Annotated Voronoi Graphs in Chapter 2, and our taxonomy of context-aware

application usage in Chapter 6. In the latter we explored different activities that context-aware applications can support. During the former we noted that different types of space can serve different purposes. In performing navigation, for example, the structure of the HAVG's upper levels support more efficient reasoning about how to obtain high-level navigation goals, whereas portions of the more detailed low-level representations are appropriate for solving more immediate navigation questions such as how to get from one side of the room to the other without crashing into a load-bearing pillar. The robot navigators access their model of the space in different ways depending on the type of activity they are performing. Our user on the other hand must navigate and interact in the space through the using a paradigm that flattens both activities to a dependence on a mutually negotiated transient proximity relationship. Whilst this mechanism—negotiated, transient, and individual—suits the process of interaction, the relationships involved in navigation are not well reflected by this form Awareness. Our changes to the platform to support rendering of spaces the user inhabits may certainly help to ameliorate this, but there remains the larger question of whether the flattening of both visibility and access into the same mediation structures that is performed by the model and platform undermines the user's understanding of the space on a deeper level. Our platform currently does not offer alternative ways to access the model, but this issue with navigation indicates that whilst presence and Awareness as they are used in the model may be a suitable envelope or framing for the sort of negotiated access and interaction as described by Dix and Chalmers & Rodden, the user may also benefit from the ability to obtain a non-negotiated view of the gamespace for the purposes of navigation. This is a process that we shall call *surveying*.

A *survey* process consists in the model of gathering the presence information but not the payload data of all objects residing at locations within a given BoundingBox. Unlike the model's existing definition of Awareness this process presents a simple frame for visibility of objects in the model. To maintain Awareness as a paradigm that concerns itself with notions of proximity in space, we name this new form Survey Awareness. Survey Awareness defines proximity as being located within a given boundary, in the

way that we can say something within our city's boundary is near compared to something located in another city. Survey Awareness is not negotiated, so it does not change when unless the user explicitly moves the boundary or changes its size and when it does change it does so in a manner that is more straightforward. We separate this type of Awareness from our model's original definition of Awareness by repositioning that as Interactional Awareness to reflect its most suitable function in the model. The Survey Awareness framing might be decided by a user by specifying a zoom level and position on a map, or it may be determined automatically by the application which, given a set of objects that the user has awareness of, wishes to retrieve all objects within that space. Recall that if objects of very large and very small nimbus both exist within the same space, when using Interactional Awareness to mediate visibility, it is possible for the user to have awareness of those distant objects with large nimbus whilst the nearer object of small nimbus remain both inaccessible and invisible to her. Through the survey process she is shown that these objects exist and, through the rendering of their aura, using the rendering service, how she can position herself to interact with them if she wishes.

In the SPACES system we propose that this service be accessed in the DataspaceConnector via two methods **`requestSurvey(String spaceID, BoundingBox frame, String userID, SurveyCallback callback);`** and **`requestSurvey(String spaceID, String[] pointEncodings, String userID, SurveyCallback callback);`** which take as their arguments a String specifying the space to perform the survey in, a BoundingBox or set of point encodings specifying the portion of the space to be surveyed, a string identifying the user that the survey is relevant to, and a callback object to receive the results. The DataspaceConnector can package these into a tuple and publish it to the dataspace for the Context-Broker to satisfy. The Context-Broker upon receiving such a request constructs and publishes a single tuple containing an array specifying the appropriate objects' presence data. The DataspaceConnector subscribes to such tuples and when they are received passes them to the application via the appropriate callback object. The same Bounding Box, or set of point encodings, as is used to construct a survey can

be passed to the space's RenderingService to produce the appropriate renderings of the survey space.

Transduction in seamful Spaces

Simondon's approach to technology is evolutive[SIMONDON'12] ; MacKenzie describes his interests as centring not on how a thing is composed but on the processes by which it comes to be [MACKENZIE'02] (p14-17). Temporal and spatial reticulation are two mechanisms by which both the evolution and the ontogeny of technologies occur, another is transduction. Simondon first describes transduction as an operation by which an activity propagates itself within a domain "through a structuration of the different zones of the area over which it operates" [SIMONDON'92] (p313). He offers the example of a crystal growing from an initial germ in a super-saturated milieu: the crystal advances via a process that progressively structures the milieu in direct contact with its borders in a manner that is driven by the existing structure of those borders. Both Mackenzie [MACKENZIE'02] (p25) and Hottois [HOTTOIS'93] (p45) present a second example of a microphone which transduces speech into electric current. Mackenzie points out that for this electrical transduction to occur there must exist some "disparity, discontinuity or mismatch within a domain" and in the process of transduction that disparity is "restructured across some interface" - the electrical signals output by the microphone are restructured to match the sounds it picks up. This is exactly what Simondon intended for transduction. Sauvanargues [SAUVANARGUES'12] and Grosz [GROSZ'12] both describe transduction as a structuring that occurs to resolve what Simondon referred to as a process of disparation between two domains - their being or becoming disparate [SIMONDON'92] (p315).

For Simondon, the concept is ultimately not just a descriptor of physical process but a tool for approaching both ontology and epistemology not as an analysis of the grounding and limits of reason or existence, but an examination of the process by which any thing, including knowledge, comes to be. This process he calls individuation. Thus, he extends the definition of transduction

to include mental processes. Bowden describes this as making transduction the process through which, within a problematic domain, a structure appears [BOWDEN'12] and Simondon describes transductive thought as being "the procedure of the mind as it discovers" [SIMONDON'05] (p32). Combes calls this instance of transduction an analogical process occurring "between the real exterior and the subject" [COMBES and LAMARRE'13] (p9). The mind recreates the exterior world in a way which mirrors it and yet envisions the resolution of its discrepancies, leading to invention and problem solving. Combes also draws attention to the human as transducer for the technical quoting Simondon as saying they "assure the function of the present, maintaining the correlation, because their life is made of rhythm of machines surrounding them which they link together." [COMBES and LAMARRE'13] (p60), [SIMONDON'12] (p136)

Transduction then can occur in a dizzying variety of situations, but all of them share some characteristics. They involve a coming together of domains, mediums or milieu between which some disparity arises and out of which some new activity, process or being generates itself (individuates) as a result of restructuring one domain to resolve the disparity. This restructuring does not have to be physical or permanent, as Grosz states it generates "momentary or longer alignments" [GROSZ'12] (p42). These disparities, discontinuities and mismatches that transduction operates on sound remarkably similar to our description of seams in fact we argue that transduction and the interfaces it occurs across are an inherent aspect of Pervasive technologies and Mixed Reality. As Mackenzie says “The hallmark of a transductive process is the intersection and knotting together of diverse realities” [MACKENZIE'02] (p13) The following sections establish the relationship of transduction to seams and its implications for our response through the model and platform, both to seams in general and to the seam of walking whilst searching specifically, which our user encountered whilst playing the Bombsquad game.

The previous section discussed the aspects of Pervasive systems that complicated their analysis and here we can see again the action of those aspects. Their situated and ensemble nature means that Pervasive applications

not only create or maintain multiple interfaces across which transduction occurs, in many cases these transductions can occur in multidimensional and unexpected ways. In the Seamful Game CYSMN, for example, we can identify the GPS location service as an interface which transduces the physical location of street runners -- tasked with catching the virtual avatars of online players -- into the mixed space of the game. Runners playing the game, however, soon discovered that there is a disparity between the structure of the mixed space of the game as shown to them and that of the space created by the varying accuracy of the GPS system. This disparity forms a seam in which some areas of what in the system's view is an undifferentiated gamespace provide different levels of location accuracy and therefore different probabilities of successfully catching an online player. The subsequent adaptation of runners to the game and the appropriation of the seam is an active restructuring of the intersectional gamespace, one which transduces the structure of the built environment -- which drives the GPS system's accuracy -- into the spatial behaviour of the players. Unlike the microphone wherein a transduction occurs across single interface and remains wholly within the designed and engineered aspects of the system, the location transduction we see in CYSMN is occurring in multiple ways and across not only technological but environmental (architectural in this case) and socio-behavioural milieu.

If seams represent sites of transduction in Pervasive systems, we might ask what prompts transduction at such a site. In CYSMN, the presence of tall buildings and narrow streets form a frame for poor locational accuracy and conversely the structure of broad and open spaces becomes a frame for the activity of chasing and reliably catching online players. What allows the CYSMN runners to perform this transduction is the highly visible nature of the framing and the strong reticulation of that framing across the gamespace for the duration of the game -- tall buildings and narrow streets consistently produce poor GPS accuracy and open spaces consistently yield more successful capture attempts. The framing creates a metastable signifier to particular behavioural strategies for runners to adopt. In Reeve's interactive storytelling installation *Journey into Space* (JiS) [REEVES, PRIDMORE et al.'06] we can see this phenomenon illustrated again. The JiS installation generates two spaces:

the space seen by the sensors, which is structured by objects, both permanent like cameras and transient like bodies -- that obscure certain lines of sight to sensors, and the space experienced by the young users who have no tools to intuit this structuration. The facilitator of the storytelling activity, who is familiar with the structuring effects of the physical space on the sensor space, brings these two spaces into alignment, restructuring the users' space through verbal commands such as "stand further back."

In other applications, the physical framing of the seam is not so strong, or is entirely absent, and a different restructuring approach must be used. Broll et al.'s Tycoon [BROLL and BENFORD'05] is an application designed to appropriate the seam of limited bandwidth, which can be viewed as a discontinuity between user behaviour -- demanding frequent updates from the server -- and infrastructural capability -- the lack and expense of bandwidth. Broll et al. restructure user expectations and behaviour by designing the game to reward those behaviours that match the capabilities of the infrastructure. Users who play entrepreneurs mining for precious metals, which can then be used to purchase objects of varying value, are told that server updates cost money and the longer they spend mining without contacting the server, the greater the amount of precious metals they will acquire. The framing here is entirely constructed by the developer and its clear and communicable narrative where mining can only occur remotely in places that are disconnected, isolated, from goings on in the rest of the world provides a strong and legible framing that invites users to restructure their behaviours to correct the discontinuity in the system. Without this framing it is easy to imagine users becoming confused and frustrated by the lack of responsiveness they experience due to slow or overloaded connections.

The seam our own user experienced bears some elements of the location and sensor accuracy issues of CYSMN and JiS and some similarities to the physically unframed infrastructural orderings of Tycoon, however we can also identify some differences. When the user, walking around the gamespace, performed a search, her current location was submitted to the context broker. There was a communication delay between the server and the mobile device,

however, and while the user waited for a response she continued walking. She was thus in an entirely different location when the search results were returned to her than that which they referred to. She found herself confused and struggling to understand how her physical location related to these results. The seam she encountered in this sequence was, similar to that managed and appropriated in Tycoon in that it was driven by a limitation of the communications infrastructure. However, it is also possessed of a temporal element. The mismatch is between the user's state as held by the server and her state in the real world, the former lagging behind the latter.

This appears to be a simple and obvious aspect of location-aware applications to anyone who is familiar with their architecture. The server is only aware of the locations the user submits to it; if she moves whilst waiting for it to respond to her, its response will be rendered inaccurate, and we might expect the attentive user to come to intuit this as did the runners of CYSMN intuit the structure of the GPS space they played in. But unlike the sensor issues in CYSMN and JiS, this disparity is not framed for our user by the physical environment. Moreover, since server response times are variable, since the user may or may not choose to move around whilst a search is completing and since she is capable of moving at a variety of speeds some of which may lead to significant disparity between her modelled and physical location and some of which may not, a disparity is not guaranteed to present itself with any regularity. Its framing is not easily apprehended nor consequently repeated. Additionally, unlike Tycoon, the disparity is not framed by the narrative of the application either, in fact the application's framing only serves to exacerbate the problem. Its episodic style of interaction with the server does not encourage the user to perform frequent searches and therefore decreases repetition of the disparity, so that even if parts of the location infrastructure were revealed to her -- for example, by providing a "submitting your current location to the server" message when performing a search -- she might not experience the seam a sufficient number of times to gain an understanding of what is happening. As discussed in the previous section, this episodic temporal structuring strips the user both of her history in the application and of a great deal of the continuity that she depends on to perform the interweaving that

mixed spaces require. One reason she has no inkling that walking whilst searching will have the effect it does is because her motion is not shown at all in the application. The searches she performs treat her movements as discontinuous events rather than slices of a continuous process. In this way the seam is not only temporal mismatch between the user's states, but a discontinuity between her continuous experience of the space and the application's discrete sampling, modelling and presentation of that experience.

The seam can be approached at the application level by showing the user's current location over the search results, but this itself creates a further discontinuity between what would be the user's awareness at her current location and the objects she has awareness of under the location submitted to the server. As an alternative, a narrative framing could be employed to draw this disparity to the user's attention, portraying the search mechanism as a camera snapshot function, as if the user is releasing a camera at her location to an aerial picture of the mixed space around her. For a technical solution in the application, we can implement a routine to detect changes in her location, whilst waiting for a search to return; one that triggers a warning "Stop moving around! You'll make the picture blurry." if her location strays too far from the location the search originated at. Alternatively, a routine could be implemented that makes the search results screen increasingly dim and out of focus as the user moves away from her search location and complains to her "We're so far from the camera I can barely see that snapshot. Do you want to take another?" Whilst the narrative and partially narrative approaches are similar to Tycoon, they in fact solve the opposite problem: that the episodic nature of the application's framing discourages regular contact with the server.

At the level of the model, however, seams are more difficult to approach. As we discussed in the previous sections, Pervasive systems are complicated by the requirement that they inscribe themselves on multiple pre-existing structures, and further by the intersectionalities of their situated and ensemble nature. We see in the applications discussed above how that intersectional nature gives rise to a seam and user response to the same which is revealed through a combination of multiple domains, not only the virtual structures of

the model and application and the physical structures of the space it acts upon, but through the temporal, the social, and emergent behavioural patterns. In moving from the specificity of user experiences in individual applications to the generality of the model and platform, it is our fear that these vital elements be lost. Our initial response in the model to the existence of seams involves using Dix's Physical-Representation-Model framing to bring forward and emphasise the differences between a user's state in these three domains and the processes of abstraction involved in moving between them. However, our user's experience with this seam indicates that this approach which divorces itself from a single specific situated application in order to provide abstract methods for reasoning about many different applications may not be sufficient to support management or appropriation of a range of seams. Our method is therefore to first explore a seams from a broader basis than our single seam supplies. To that end we now present an analysis of various seams found in the literature, their transductive properties, the framings arising with or enacted upon them and possible alternatives to these framings.

Transductive Properties and Framing of Individual Seams

As previously discussed, the disparity in CYSMN occurs between the unstructured system space and the GPS representation space which is structured by its sensors' interaction with the physical environment. This structuration of the GPS representation is framed and presented to users through patterns they encounter in the built environment. The transduction that users enact at this seam is performed autonomously and socially, which is to say they are not directed to perform it by the system or any third party and their performance of it involves a measure of communication and collaboration, sharing information about which spaces are good for pursuit and capture and which are not. An alternative, or additional framing for the seam could be constructed within the application by displaying onscreen an estimated location accuracy (dependent on the number of GPS satellites visible to the device, for

example). This would allow users to see when they are in areas of poor signal quality, but it is not a substitute for the metastable framing of the physical environment whose reticulation allows them to predict a space's location accuracy before they enter it. If we could take the physical features themselves away from the gamespace, yet retain their effects on the GPS representation, users would require not just an indication of their GPS accuracy, but a map showing zones of accuracy vs. inaccuracy to support their appropriation of this seam to the same extent that their experience in the physical environment does. This is MacKenzie's ensemble and situated view of Pervasive systems in action. The city's unique structure gives structure to the application that can only be understood by users through their experience of both city and application in tandem.

There are, however, alternative routes to transduction across the seam. Mapping and making available to the user the structuration of the sensor representation is one approach to a situation where that structure is not reflected in the gamespace model, but a developer might also restructure the gamespace to exclude areas of inaccurate location sensing from play, thus removing the structuration of the representation and aligning it with the unstructured gamespace. Alternatively, they might restructure the gamespace's rules concerning how close a runner must get to an avatar in order to capture them when in an area of low location accuracy. Both of these approaches are predicated on the developer's ability to detect and isolate the disparity between the representation and the gamespace model, if not correct it. And both of them involve transduction enacted as a managing of that disparity on the user's behalf, rather than enacted by the user as an appropriation of the seam.

In Journey into Space (JiS), the disparity is again between the structured space of the representation and the unstructured space that the application presents to the user. The framing that emerges here, too, is a result of the physical structure of the space and additionally this time the repetition of certain transient spatial configurations, such as a child standing so close to the activity wall as to block sensors with her body. Instead of arising in the behaviour of the children, the transduction is a restructuring of the space performed through

corrective commands given to them by a professional storyteller and activity facilitator who has developed enough familiarity with the installation to recognise the framings of its seams. An alternative or supporting framing here would be to mark the floor to delineate places for users to stand that do not position them in conflict with the spatial configuration of the sensors.

The children experiencing JiS are not aware of the seam in a similar manner to the way that CYSMN's online players are unaware they are playing in a structured gamespace. In the child's case this is not because they do not directly experience the seam but because the storyteller manages the seam for them; the application adopts this facilitator as a part of its transductive infrastructure. Similarly, Tycoon's seam has no direct exposure to the users but is framed and managed for them using the external narrative of the game. This narrative acts upon the mismatch between demands users might make on the game and available resources and restructures user behaviour to match the infrastructural resources available to the app. The disparity, however, remains veiled to the user rather than explicitly framed. An alternative, explicit framing might present hard and visible limits to the user on the number of times they could contact the server for an update, information about the speed of their connection, and the last known server load.

Users in Savannah experienced two different and somewhat more complex seams, which were not transduced by users or the application. The first was a mismatch between the unstructured continuity of the physical space and the discrete structuration of the gamespace, complicated by collaboration aspects which required the users, playing the part of hungry lions, to construct a socially shared view of the mixed space with their teammates in order to coordinate hunting attacks on large prey. This disparity, driven by the attempt to virtually structure a physical unstructured (or differently structured) space, contrasts with the seam in CYSMN wherein a physical space acts to structure the user's experience of a virtual or mixed space and does so in a non-explicit manner. The second seam Savannah's users encountered was GPS sensor drift, which caused a temporally complicated disparity between the user's physical state and her state in the application where due to small changes in GPS

reading, she appeared to be moving when she was standing still. This seam also interacted with the first, so that users on the boundary of areas where they could attack prey would, without moving, suddenly find they were no longer close enough to perform an attack. Since the app was not designed to explore seam appropriation, neither of the seams were given framing in the application. Once possible framing of the first seam would be to present users with some of the virtual boundaries in the game and allow them to see where both they and their teammates are in relation to these. This is a similar approach to that of mapping areas of differing location accuracy in CYSMN in that it brings forward a previously obscured structuring of the space, albeit one that originates in the application rather than the intersection between infrastructure and environment.

Transduction of the seam with this framing could then be performed by users who would now have more information about the structure of the mixed space. An alternative, as mooted by Savannah's developers, uses mechanisms to extend areas for attacking prey animals slightly to encompass the entire team when one member is within the area. This restructures the existent structuration of the virtual space without presenting any of those structures to the user, managing the seam for her. The second seam differs from those in CYSMN and JiS because although it is a sensor-based seam, it arises not in the interaction of the sensor with the environment, which is easily framed for user appropriation, but from the technicity of the sensor itself, neither does it lend itself to transduction via a narrative framing as in Tycoon, as narrative framings act directly to restructure user behaviour and there is no useful restructuring of user behaviour that can be achieved in this circumstance. The only reasonable option left to a developer is to hide and manage the disparity of the user's behalf. For example, when position updates fall below a certain threshold of difference, the system might switch to taking the user's location as an average of locations sampled over the past few seconds. Seams with a similar technical origin appear in applications that use GSM and Wi-Fi positioning, for example: the effect of weather conditions on signal propagation and the phenomenon of cell location "flipping" where a stationary

device situated on the border of two GSM cells experiences continual cycling of its location from one cell ID to the other and back.

One of the most complex processes of transduction we encounter in the literature comes in Chalmers Mack Room application [CHALMERS and GALANI'04] , which mixes situated, 2D online and 3D VR user experiences of a museum exhibition. Disparities arise in multiple places driven by how users of the different technologies were able to access exhibition content, what content they could access and the methods by which they viewed and navigated the space. The application can be thought of as maintaining multiple spaces with disparities of structure, behaviour and information between each of these. The interface between these spaces is the mixed or aggregate space itself that arises from their intersection and a successful application is the result of constructing reorderings across the boundaries between these spaces such that users can enjoy the museum exhibition as a group.

The three users of Mack Room collaborated to socially and cognitively restructure their respective spaces so that their experience of the exhibition instead of being separate and disparate became a mutual and coherent process. The disparities were framed through two mechanisms. First developers provided a visual indication of where the other users were located in the mixed space of the application, whether that location was drawn from mouse clicks on a map (online), the position of an avatar in a 3D model (VR), or a sensed physical location (situated). The second framing was an audio feed which provided the users with a venue for interaction. Additional framing could be said to derive from pieces of information about the exhibition which were shared across all spaces. These were leveraged as anchor or orientation points by users who frequently emphasised and repeated them in order to support the restructuring of each other's individual view of their respective spaces to a view structured instead by these common landmarks. In particular, Chalmers work here demonstrates that users are capable of performing very complex and autonomous and indeed asynchronous acts of social transduction given framings that support reticulation through communication. Users were seen to break from the group to explore their own space and later re-join the

group's activities by requesting over the audio channel information that would help them reorient in the shared space by repeating activities (visiting exhibits) that the group had performed without them. The audio stream and overlapping features of the spaces intersected to create framing which encouraged not only repeated acts of verbally emphasising common landmarks, but moreover constructing them, for example labelling a curved wall as the "boomerang wall" and repeatedly referring back to the label in order to coordinate activities in the space. This powerful framing could be the result of the juxtaposition of the audio stream and shared user location streams, with the pieces of information or features, linked to the situated exhibits, shared across the three spaces. The first two form a framing that works as a type of shared intensive history, encouraging both dispersal and repetition of the transductive act of naming shared features. Through this, the latter having both a set location and long-term persistence are available to be positioned as features of a shared extensive history in the exhibition space. Similar capabilities in other applications might allow users to describe, annotate or otherwise share and define the structure of the spaces they move through with others in ways that encourage both immediate interaction and a metastable persistence by retaining those features as landmarks in the system.

Conclusions about Seams

Transduction

Transduction refers to a general process of restructuring one or more aspects of a system to resolve disparity between them and thus can operate in Pervasive systems in myriad ways. Although it acts across seams, we can see in our survey and analysis of individual seams from the literature that it is not in itself necessarily seamful as Chalmers, MacColl and Benford define seamfulness. The transductive action of averaging a user’s location readings to eliminate drift, for example, does not exploit the disparity and therefore is not seamful. Nor can we say transduction always eliminates seams or gives the appearance of eliminating them. What transduction refers to then is not seamfulness, but any way in which an infrastructure, application or user engages with a seam

which involves a restructuring of one or more elements associated with the seam and that engagement may, as in CYSMN and Tycoon, be seamful.

These restructurings can occur upon user behaviours, upon any of the spaces involved in the application including spaces that users construct cognitively and/or socially, upon user states, and upon infrastructure as a non-exhaustive list. Chalmers' Mack Room illustrates the complexity that can be involved in processes of restructuring that occur socially and cognitively. The extensiveness and complexity of transduction's reach here demonstrates the same ensemble and intersectional complexity of which Pervasive systems are themselves possessed, and which we have discussed previously. As our focus here is on changes we might make to our model and platform, however, the performance of these complex restructurings lies naturally outside our scope.

With respect to less complex transductions, a platform can support re-orderings of its modelspace, of its representations, and of its own structure and processes. The easiest transductions to think about are those a system performs to hide disparities from the user. Our suggestions that CYSMN could remove areas of low location accuracy from the gamespace or relax the required catching proximity within those areas are two such transductions that change respectively the modelspace and the processes of the application. These sorts of modelspace changes are already somewhat supported by our SPACES model and platform through their mechanism for updating the representations of spaces.

Managing Bandwidth Related Seams by Moving Computation

Another seam which commonly affects mobile applications is the speed of their device's data connection. Applications commonly manage this seam by caching information they expect the user to require in the near future; this too is a transduction and one which is well understood by developers. The disparity being engaged is between the user's behaviour and the capabilities of the infrastructure supporting the application, it takes time to fetch information, the user does not want to wait. In response, the infrastructure is reordered so that information is moved closer to the application, in easy and immediate reach. But it is not just information that can be relocated within an infrastructure. We

also have the architectural choice to move parts of the model's execution closer to the user. A reordering of this nature can be performed in our platform by allowing the context-broker to return partially computed awareness information so that the client application can perform the final resolution with respect to the user's awareness of individual objects. This gives the system the capability to manage and hide the Walking Whilst Searching seam our user encountered during the application's trial. Since under these conditions, final determination of the user's Awareness will be deferred until that information is on the device and about to be displayed, the application is able to sample and use a more current presence reading to derive her final Awareness than that which was originally provided to the context-broker.

Since computation of Awareness can span multiple spaces, including spaces of which an application has no knowledge, this relocation of the evaluation of context is not as straightforward as simply allowing an application to compute the user's Awareness values from her and other entities' presence. This would require not only that the application understand spaces that it does not manage, but that it also understand how spaces are related to each other and maintain a duplicate of the model, or at least part of it, as held in the Context-Broker. Furthermore, since computation is being moved in order to overcome a slow data connection between the mobile client and server, the computation must be performed not on the server but on the client, potentially requiring that the client perform spatial reasoning functions supported by the Context-Broker which we designated as a central service for context-reasoning in part because we did not wish to place the processing burdens on mobile clients.

Instead, in order to achieve the responsiveness, we desire, we propose a change to the model and platform that allows a Consumer to present either a single presence update, or an array of possible presence updates representing a range of discrete values that the user's presence might take in the near future. We say that each presence in this array generates its own Awareness set which is the Awareness of Producers in the space a Consumer would have that particular value of presence. In the platform, when presented with this array of presence values, the Context-Broker calculates the Awareness results set for each value.

As the platform currently operates, when the Context-Broker computes a single presence update each Payload that the user is to receive is published into the Dataspace in its own tuple with its Producer's presence information. In our update to the platform, we propose to introduce a new object called PayloadIndex which stores a collection of Payloads from a single Producer relating to the presence array and permits access to any specific Payload using its associated presence value (Location and Aura) as a key. Since the Context-Broker is computing these sets of Awarenesses, the application does not need to know anything about other spaces in the model and in particular does not need to understand the details of any presence the user may have in another space. Each Payload, whether it originates in the user's native space or has been gathered from an object in a foreign space she has been projected into, is accessed by the value of a presence that the user has in her native space.

The final step in the process is to move the information across the seam of the slow data connection to the client, so that it may select the most appropriate Awareness set given the user's most current presence value. The information from the PayloadIndex can be extracted, serialised and returned by the server application to the mobile Bombsquad client which can then select the results for the presence value that best represent its user's most current presence. When submitting a presence update, (performing a search in our Bombsquad application), the array of possible presence values can be generated in the mobile client or server application by examining trajectories and speeds that the user is travelling at when she makes a search request. In this way a user who performs a search whilst she is still moving can be presented with a more accurate set of results than the seam originally allowed for.

Framing

In contrast to transduction, seam framings are relatively simple to think about. Unlike transduction which is a process, framing acts merely as a presentation of some aspect of the system to be transduced or as a venue for that transduction. Framing has a much closer relationship to seamfulness than does transduction in that all seamful applications – those which exploit seams in their design – perform some sort of framing. It is tempting to push this further

and say seamful applications all enact a framing that presents one or more of their seams to users and that all applications that perform framing are seamful. We believe, however, that this is not the case. CYSMN, for example, was not explicitly designed to exploit the seam its users appropriate and does not qualify as seamful by that particular criteria, yet it possesses framing, transduction and (autonomous) user appropriation of a seam. Narrative-type framings in particular seem to straddle the various categories of seamfulness in untidy ways. Broll et al. characterise Tycoon's narrative framing as seamful because it employs an infrastructural limitation in an aspect of gameplay. But although Tycoon is seamful in that it is designed to exploit a seam through user transduction, we would not say its framing presents that seam to users. Rather, since our transductive approach to seams characterises them in terms of disparities, the framing creates a lure to a particular sort of user behaviour and in doing so acts to disguise or direct attention away from the disparity that exists between usual user behaviour and the limitations of the infrastructure. Framing then is a signal that users are involved in the transduction of a seam but not a specification of how that involvement proceeds, nor the seamfulness of the application.

A platform developer is only of course able to consider what framings might be possible within the platform and how those might be achieved which rules out narrative framing as a device that she must consider. Physical environment framings are usually, though not always, out of even the application developer's control to create or modify—the developers of CYSMN for example have no control over the built environment's structure. In an installation like JiS, a physical framing can be deliberately introduced to help users develop an understanding of the space's structures in the same way that buildings do in CYSMN. Not all applications offer this opportunity, however, and the creation of physical environment framings in any case are certainly outside the control of our model and platform.

Framings that bring forward aspects of the sensed and constructed space, however, are possible in both the model and the platform. These can also be employed as transductions that actually restructure the model space in certain

ways. For example, annotating a GPS space with predicted accuracy values and displaying the relevant values to users acts as a restructuring itself of the representation space and also constitutes a frame for further transduction with respect to user behaviour. Annotating the user's current location within the space with a certainty value, on the other hand, would simply frame the disparity so that he might perform the transductive restructuring of his behaviour himself.

The change we propose to our model is one which introduces structurations to the model as layers of annotation over a particular space. Since each space is defined by a set of points and structurations reorder these points in some way, we define a structuration to be a set of annotations paired with individual points. A space may have zero or more uniquely named structurations active upon it.

We define a new generic class Structuration in the Platform, which is to be instantiated by any application wishing to maintain a structuration of the appropriate space and passed to the context-broker within the space's definition tuple. The Structuration class maintains a collection of annotation pairs, which consists of pairings of a label, which may be a string, Boolean, or number, with a Point object. We propose that the Structuration class provide an abstract method getSubStructuration(BoundingBox box) which given a BoundingBox that defines a subset of the space that the structuration orders, will return a SubStructuration object which wraps a subset of the appropriate annotation-point pairs. When this SubStructuration is returned to the requesting party, these values can be accessed through SubStructuration's enumeration method getNext() which returns the next annotation-point pair in the set. To support Structurations in generating Substructuration objects, we add **`a public boolean contains(Point point)`** method to the BoundingBox interface. Given a Point in the space, this, should return a Boolean value specifying whether the Point is bounded by the Box or not.

Since the activity of Survey, previously described, is concerned with retrieving information about the space which is mediated by the paradigm of Survey Awareness, we make the retrieval of the space's structuration(s) part of this

activity. When an application wishes to retrieve a portion of the space's structuration it can submit a Survey request for the space, which will by the previous definition of the Survey request, include a BoundingBox object. This BoundingBox can be used to define the area that the sub-structuration returned should cover. To satisfy the structuration part of such a request, the context broker supplies the space's Structuration object(s) with the BoundingBox and receives in return a SubStructuration object which it publishes to the dataspace along with the Survey response tuples.

Mack Room and Savannah both illustrate the importance of framings that support user communication and collaboration. Framings that make user behaviour visible and accessible to other users as were enacted in Mack Room, constitute part of this. We discussed earlier in this chapter the need to provide different views of the space – different contexts of Awareness– depending on whether the user's activity could be characterised as gathering or navigation. To support applications in displaying the sort of real-time framing of the space that is used in Mack Room to share users' locations, we propose to extend the Survey process both to include displaying the locations of Consumers and to respond to any changes in presence of entities in the model by forcing an update to the published Survey results set.

Our model is designed around a Producer/Consumer interaction paradigm. Recall that Consumers are entities which have Focus (area of interest) but not Nimbus (area of influence), in the model, whereas Producers who generate information payloads have Nimbus but no Focus. This change is in effect a bestowing of a type of influence in the model upon Consumers, but influence of a very particular sort. They are no longer invisible to each other, which supports the sort of social transduction that Mack Room fostered, however, in comparison with interactional Awareness the framing within which they are revealed remains relatively more stable as they move through the space. If a Consumer wishes to produce information for other entities to consume, however, for example an audio feed, she must still explicitly situate herself in the model as a Producer with an additional point of presence and Nimbus Aura. Since Producers and Consumers are possessed of the same types of location in

the platform this change to the Survey process can be implemented in the Context-Broker simply by adding the Consumer presences list that it maintains to the list of entity locations it will assess when constructing the Survey response.

When discussing the Survey process previously, we made no mention of what happens in the platform when the presence of some other entity within the survey space changes. The assumption was that Surveying is an information-pull process rather than one in which updates are automatically pushed to the subscribing application. This creates a more episodic framing for the act of Surveying and we have previously discussed the usability shortcomings in such an approach with respect to navigation-related tasks. We therefore propose that applications are able to subscribe a user to a Survey over a particular space and receive not only a single set of presence information for entities contained in the BoundingBox subspace, but also updates to that set whenever the presence information of an entity within that subspace changes. This can be achieved by adding routines to check Survey results to the main routine that runs whenever an entity update is received by the space. The routine extracts the location from the updated entity and checks it against the BoundingBox currently associated with each survey, triggering a tuple add or update event for that Survey result set if it finds a match.

Finally, with respect to framings, we have already discussed how making a user's history – both intensive and extensive – available to her can improve the legibility of the model space. In Mack Room the visitors' use of repetition and continued return to stable anchor features also indicates that a historical framing – a bringing forward of some stable and persistent features of the model, or repeating of older features -- highlighting objects she has already interacted with, for example -- can support the user not only in understanding the model space but for example in transducing disparities between her view of the model and an another user's to support collaboration. The question of how this might be implemented in the model and platform, by returning with each payload a history of the user's interaction with the Producer that generated it, was discussed in previous sections.

Conclusion

In this chapter we analysed our model, platform and application with respect to three issues that our user encountered during a trial of the SPACES platform through the Bombsquad game. The issues affected the user's ability to read and construct the mixed space she found herself in and due to the complicated nature of those activities we sought out approaches to the construction of meaning from space and technology that have been developed in other fields. By bringing some of the concepts from Gilbert Simondon's philosophy of technology and the field of Architecture to bear upon the questions these issues raised, we were able to identify the elements that affect the legibility of our system and offer suggestions for correcting this at the level of the application, the platform and the model. Our approach also leads to what we believe to be a novel and useful way to approach seams in Pervasive Computing systems, not as technological uncertainties or sources of error, but as disparities between technological, social, temporal and environmental elements which may be resolved through processes of reordering enacted by applications, developers or users themselves. In Chapter 3 we determined the importance of continuity of a space to users' understanding of that space, a position that drove our adoption of quantitative awareness alongside qualitative awareness to smooth the transitions between differing levels of awareness. Our proposed changes to the application involve ensuring that continuity is maintained temporally as well as spatially by providing the user with continual feedback about her movements in the model and a history of her interaction with other entities.

In addition to this, we identified a failure to make visible and bring forward the structures of the model, the physical and the mixed seamful space as contributing to user confusion both in our application and in work by other researchers concerning seams. To combat this, we proposed alterations to the application and platform which would make presence information more visible to users and support applications in rendering the physical structure of spaces, as well as modifications to the platform to support it in maintaining structurations of the model's spaces and presenting them to the user as a framing for user transduction of application seams.

Finally, our analysis helped us to recognise that Awareness, as an individually negotiated and transient mediation infrastructure, is not always the appropriate framing for user behaviour. We introduced a Survey function which acts as a window on a subset of the model, providing Awareness information mediated by proximity as defined by presence within a boundary rather than the interaction of areas of Interest and Influence. We believe for the purposes of mixed-space legibility, the visibility of objects in the model requires a more stable and repeatable than the Interactional Awareness framing offers and that this Survey process provides that.

In the next Chapter we offer our conclusions on the work as a whole and suggestions for future research to be undertaken in this area.

Glossary of New Terms

- **Intensive History** A form of user history that is cognitively maintained by the user herself. It is rich in her fully lived experience but persists in this richness only for a short period of time
- **Extensive History** A form of user history maintained by the application which is necessarily not so rich as the user's personal experience but is transmissible, durable and can be made available to other users and applications.
- **Survey Awareness** Used for activities where a simple view of the space is required. Defines proximity as being located within a given boundary rather than dependent on the negotiated scale of Nimbus/Focus overlap.
- **Interactional Awareness** The awareness measure that our model initially used as defined in chapters 3 and 4, which is dependent on the negotiated scale of overlap between Consumer Focus and Producer Nimbus. Renamed here to help differentiate it from Survey Awareness.
- **Transduction** A process that occurs across domains, mediums or milieu between which some disparity has arisen and out of which some new activity, process or being generates itself as the result of restructuring one of the involved domains to resolve the disparity.
- **Framing** The presentation of a space, seam or process which brings forward certain of its properties or aspects, conditioning the user's understanding of it and response to it.

Chapter 8

Conclusions

Introduction

A user's context can be formed in many different ways and comprised of types of data whose structure is still to be discovered or changes during the course of an application's runtime. Not only is context heterogeneous and dynamic, new technologies that sense or generate new forms of context are created regularly. Developers of Context-Aware applications can find it difficult to support and reason about all these forms. Moreover, as a type of Pervasive and often Mixed-Reality computing, Context-Aware systems frequently require developers to think about complex intersections between the physical environment, virtual elements, infrastructural capabilities, and social and historical processes enacted by their users. These intersections may also form the basis for seams – those sites of interface between two elements of a system that give rise to often problematic disparities. These issues set the parameters of our research.

Aims

The aims of this work were primarily to develop a framework for thinking about context that can support a range of different types of contexts, and to investigate how we might support applications in reasoning about contexts both those that they understand and those that are new to them. Our secondary concern was to offer ways to think about and analyse Context-Aware systems explicitly in terms of the intersections and seams that characterise Mixed-Reality. Since seams are a significant part of context-aware applications and often under-engaged at the platform level, we also wished to enable developers to create applications that are more capable of managing seams or presenting them to users for appropriation.

Methodology

We pursued our research aims first by constructing a way to model different types of context and the states of users possessing context of those types. As analysing Context-Aware applications revealed a pattern of interaction between proximate entities, we chose to employ Dix's Physical World-Representation-Model architecture and the presence and Awareness archetype of Benford et al. as our model's foundation. The model was realised in a Spatial presence and Awareness Context Evaluation Service (SPACES), which developed on top of the Equip Dataspace, consisting of a Context-Broker which performed reasoning about user presence and Awareness within and across different context representations, and an API which provides the structures and processes for supplying the Context-Broker with the information it needs via the Dataspace.

The centralised Context-Broker both takes the burden of computing Awareness per the standards of the Model away from applications and offers opportunity to gather in one place a large volume of information about the relationship between context representations via concurrent user updates. This gathering of spaces and their presence updates to one centralised service thus supports our aim to allow users possessing one form of context to gain awareness of those possessing another form of the same type of context.

The platform and model were tested using a simple game whose gameplay was designed to involve or support the most common aspects of context-aware application as we determined them from our analysis of the literature. Issues that arose during a trial of the game then directed our analysis of the systems' problematic aspects and its relationship to seams. Since we felt our experience indicated the difficulty of applying traditional methods of analysis to Pervasive Systems, especially in the presence of seams, we made the choice to employ alternative paradigms drawn from the fields of philosophy of architecture and technology: framing, reticulation and transduction.

Research Questions

We began with 6 questions about reasoning with context. The following summarises the ways in which these were answered in the course of our work.

Is there a way to think and reason uniformly about context that supports the heterogeneous types and forms of context that are found in context-aware applications today and may emerge in the future?

Chapter 3 puts forward a conceptual model for reasoning about context, using the system of presence and awareness, which supports a variety of different types of context through its abstraction processes and four types of space: Continuous Symmetric, Continuous Asymmetric, Discrete Symmetric and Discrete Asymmetric. Our platform's API given in Chapter 5 offers the flexibility for developers to introduce new types of context through their implementations of the abstract classes that define the different types of space and their structures.

How can we describe context in representations whose content and structure are still being discovered, or may be subject to change, in such a way that we retain the ability to reason about those forms of context?

Chapters 3 and 4 present the ways in which our model supports context representations whose full structure is not yet known or is subject to change. The distance metric, and edge structure in the case of structured discrete spaces, support the specification of relationships between individual points rather than across an entire space so that relationships that some reasoning can take place even when relationships are not already known. The projection point relationship supports the relating of context elements at a point-to-point level, without the requirement that we understand how the entirety of each space, which may not be known, relates to the other. They also support runtime building and editing of these relationships as is described in the processes of our Context-Broker in Chapter 5.

What structures and processes in a Pervasive Mixed-Reality application drive its ability to be meaningfully understood by its users?

Chapter 7 analyses the user's experience with our application and platform through the lenses of architectural theory and philosophy. It uses theories from these to tease out the structures – framings – and processes –metastability, reticulation and user activity– that are important to the user's understanding of the application.

How can we think about seams in terms of these structures and processes at an abstract level?

Our discussion in Chapter 7 of transduction—a process of restructuring one or more elements to remove a disparity across the interface between them—offers us a way to think about seams abstractly as points of disparity across interfaces which can be resolved through a restructuring that is performed technologically in the application narratively by developers or cognitively and socially by users.

How can those elements be employed by developers to create platforms and models whose structures are more easily comprehended?

Chapter 7 proposes several changes to the structures and processes of our platform and model that our analysis indicates will support users in better understanding the presence and awareness model as a paradigm for context. Providing access to user history, increasing the temporal continuity of the application and making visible the virtual structures of the space, encourage and support the user in performing the sort of temporal and spatial reticulation of signification pairings that increases her appreciation of the model's meaning. Offering separate framings, separate paradigms of awareness—interaction and survey—to support different user activities—gathering & interaction, and navigation—within the model is another strategy we present for increasing usability of the model.

How can those elements be used by platforms in order to provide services that allow developers to mindfully manage, present or take advantage of seams?

Our discussion of transduction in Chapter 7 led to the conclusion that the most important factor in supporting developers to manage, present or exploit seams is making visible the disparities of structure between different elements. We note that this can be done in a variety of ways, including through modifications to the physical environment, the construction of application narratives and technical approaches in software. In our model and platform, we demonstrate the latter through our proposed addition of space structurations which are designed to record and reveal the hidden structures often imposed on virtual and sensed spaces in Pervasive applications. We additionally concluded that management of seams generated by infrastructural issues such as latency, bandwidth and processing limitations are best supported by being flexible about where the execution of logic takes place—restructuring the application. The mobile, distributed nature of our application puts introduces both latency and processing pressures and in a trade-off between these limitations, we propose a change to the platform and model to offer optional reasoning on arrays of location and aura values rather than single presence points. These allow applications to retrieve a range of payload values from a Producer to a mobile device and select for the most appropriate value using the location and aura closest to the user's current state.

Contribution

This work offers the reader two ways to think about Context and Context-Aware applications. The first cuts across the technological end of the spectrum, positioning context as spatial in spirit even where it is not so in actuality, and offering, through the SPACES model and platform, a mathematical approach to reasoning about it as a spatial relation that reduces its evaluation to that of a handful of inequalities and simple equations referring to proximity of pairs of entities within and across spaces of four different structures.

The second, less tidy, way to think about Context-Aware applications is as part of a system of framing and transduction whose structures perform a gathering

of entities under certain criteria within a space and frequently offer themselves either to a developer or to users for processes of restructuring that range from those of strictly defined algorithms to the more complex situated behaviours of users and in the case of applications like MAC Room: groups of users.

Conclusions

Implications of the SPACES model for Context-Aware application developers

Our work shows that it is both desirable and possible to identify structures and properties common to many the different types of context and, using these, abstract those types of context so that they may be reasoned on in a uniform way. Our SPACES model provides a way to do this. The work also shows that although proximity is a fundamental aspect of context, developers employing it should be mindful of other spatial criteria such as containment which can also be used to reason about context. In addition to which criteria is selected, of equal importance is how this criteria is made available to users. Our findings were that criteria should vary dependent on the sort of activities it is to support for example using Interactional Awareness for gathering objects, but Survey Awareness for tasks like navigating towards a specific object. How those activities are framed by the client application is also of great importance as this is frequently what communicates the criteria of the context reasoning to the user. What we have termed Interactional Awareness in our platform acts on criteria of proximity over scale that is negotiated between user (Consumer) and object (Producer), whereas Survey Awareness is defined through criteria of containment within a specified boundary. In our trial, we saw the user struggle with navigation tasks partly because of the flattening of the space's structure through mediation on a single criteria: negotiated-scale proximity, and partly because that Interactional Awareness criteria which depended on the interaction between the user's focus and object nimbuses of varying sizes, was not adequately presented to the user by visually illustrating the extent of object Auras in the mobile client's search results or by indicating where new objects had entered the user's awareness using cues like variable icon sizes.

We also saw interactions between the situated infrastructure of the application – specifically the bandwidth available to the user over her mobile network – and the platform's processes – the calculation of Awareness in the Context-Broker rather than on the device – which gave rise to a seam that acted as another source of confusion for the user. This experience illustrates that even at the level of the theoretic model with its many abstractions, Context-Aware and Pervasive developers cannot isolate themselves from the situated and ensemble nature of their applications and must consider the details of those aspects in their model's design.

Questions that developers may find useful to ask about their Context-Aware applications when following the SPACES model are: What are the properties of the space this form of context generates? Are its states continuous or discrete? How is distance between different states of context determined? What sort of context-supported activities are involved in this application? (of which our work identifies Notification, Gathering, Interaction and Navigation) What sort of criteria of proximity should be employed in the application – Interactional or Survey are two we identified. And how does the framing of the context criteria in the application support the user's activities? We discuss questions relevant to Seams and situated application development in the section Implications of Transduction and Framing for Mixed-Reality.

Implications of Transduction and Framing for Context

Transduction is a useful way to think about seams but it also has a wider implication for developers of Context-Aware applications in that it can be applied to context to allow us to think about such applications as enacting restructurings of some of their elements within the frame of context such that a particular frame, i.e. Context, indicates a particular restructuring.

Contexts that can be positioned as spatial relations in the ways we described in the model are particularly suited to this way of thinking. Interactional Awareness can then be thought of in terms of a multi-dimensional restructuring on the space, one which brings forward a subset of its entities to degrees which vary dependent on the user's quantitative awareness of them and with valences

that vary according to her qualitative awareness. Its criteria too, are multidimensional, Awareness being dependent not just one the location of the object, nor its location in relation to the user, but its location in relation to the user and that distance in relation to the sizes of their Aura. Survey Awareness provides a different, more simple, restructuring, one which brings forward the entities within a certain area. The way in which an entity is brought forward also differs from that of Interactional Awareness. An entity is either visible because it is within the area or it is not because it is located outside of it, there is no valence or spectrum to that visibility.

Useful questions for developers to ask from this position are: What are the elements undergoing restructuring in the context evaluation? What is an appropriate restructuring for these elements? Does this restructuring have a depth or additional dimension to it like that of Interactional Awareness? Is it a binary structuring like that of Survey Awareness? How does their structure appear before and after the process? How does this restructuring resolve the disparity between elements? Is the restructuring framed with the user's context in such a way that she understands that the process is taking place and has the tools to repeat it? In particular, where a restructuring operates on and/or produces multiple dimensions like that of Interactional Awareness, is the user being made adequately aware of all the dimensions that the restructuring's criteria acts upon?

For SPACES under the Bombsquad application, the elements undergoing restructuring are the clues, bombs and landmarks positioning in the gamespace. Before the process occurs, their structure is their position in the gamespace, but after a search, they have been restructured into a subset of objects accessible to the user in varying degrees upon a framing that is comprised of the user's location in relation to the object's and the size of both her focus and the object's nimbus. The object's nimbus is neither under the user's control, nor visible to her in the application. This makes it difficult for her to gain understanding of the restructuring through its temporal and spatial reticulation.

Implications of Transduction and Framing for Mixed-Reality

Transduction reaches similarly into the Mixed-Reality aspects of Context-Aware applications. Where applications create sites of interface between virtual elements and physical spaces, Chalmers' findings suggest users will aggregate these elements through a process of interweaving. This interweaving, especially where elements have differences in structure, can be considered to itself be a process of transduction – a restructuring of one or more of the elements involved in order to resolve the disparity between them through processes such as bringing forward shared features, emphasising features of the elements that are ordinarily not emphasised, and positioning features as equivalent or related to each other.

With this approach to Mixed-Reality in mind, questions that developers should ask about their applications, which in particular may help them avoid the issues we saw arise with SPACES and Bombsquad, are: What are the sites of interface between physical and non-physical elements? How do their structures differ at these sites? Is there a structuration in the virtual space that we are trying to bring into the physical space in some way? For example, having to stand in a particular area to see or access an object. Is there any other structuration enforced on the virtual or physical spaces by characteristics of the sensing technologies? Are the disparities between these structures managed by the application or by the user and if they are to be managed by the user, what restructuring is it we expect her to perform, and is she presented with sufficient information about the structurations to perform it? Finally, is she also presented with a sufficiently visible framing for the restructuring and one that is duplicated across the space and repeated through time with enough stability and frequency for her to understand through it the process of restructuring?

Future work

The temporal in transduction

A complicating factor of transduction is that disparities may possess a temporal component. For example, the way in which our Walking Whilst Searching

seam emerges as a pattern of the user's state in the model lagging behind her physical state, or the constant small yet frequently problematic changes in location of stationary users caused by GPS drift in Savannah. There is a sense that transductions involving these sorts of disparities are different because time, unlike space, is not so easily available to us. We can't change the past, nor see with certainty the future, so transduction becomes a bringing of the past into the present as in averaging GPS readings to combat drift, a reordering of repeated sequences of actions such as our proposed changes to Bombsquad enacts on user behaviour by making search results inaccessible if she moves too far from her location, or a guessing at which events may occur in the future as in the proposed changes to SPACES that allow the Bombsquad application to move choosing of awareness values onto the mobile client – a bringing of possible futures into the present. The reorderings of space we have discussed feel closer to our treatments of the temporal in user History – that History taking the shape of a space that being fully known and understood, we may then restructure in response to the user's Awareness, pulling forward certain events within it, such as highlighting the times a user has previously encountered an object.

We feel a more in depth investigation of the disparities with temporal aspects that can arise in Mixed-Reality applications, of the sorts of transductive processes they involve, and of the sorts of framings that can be used to support those processes, could provide useful information for developers seeking to both manage seams in their applications and present them to users for appropriation.

Projection Points and Reasoning across Representations

One aspect of the model we did not get to explore in the Bombsquad application is the effectiveness of Projection Points and their influence on the model's legibility. These, too, act as a transduction within the model that can be thought of as the application's own version of the interweaving users perform on Mixed-Reality spaces: a restructuring of different representations

of the same types of context to bring them into a form of alignment with each other so that users may gain presence in representation spaces that they would not normally be able to access. That our user experienced difficulty understanding her relationship to the model space when the platform was employing just a native presence for her, implies that she may have further difficulty understanding how her physical presence relates to presence across multiple representations and it may be necessary to find ways of presenting these as a single presence in one aggregate space whose rendering is constructed and served by the Context-Broker.

Exploring individual properties supported by differing types of space

Chalmers and MacColl hold that where elements of computer systems differ in their capabilities, an approach to managing those differences that flattens them to their lowest common denominator is one that does not do justice to the capabilities of the system [CHALMERS and MACCOLL'03] . It is much more desirable, though also difficult, he says to allow things to be themselves. In our model, we abstract spaces that support different types of reasoning about spatial configurations so that a single simple reasoning process may be employed across them. Although this simplifies the process of reasoning about Awareness greatly, it leads, for example, to a situation where although a Continuous, Symmetric space that uses a Euclidean distance metric is able to reason about properties such as direction, the SPACES model and platform do not take advantage of that capability. This is because when users are projected into such spaces from representations that do not support such properties it is difficult to decide how to construct and reason about their projected presence. We believe an investigation into how we can reason across context representation spaces in a way that allows us to preserve these properties and relationships where available would be a useful extension to our model.

Structuration affecting presence and Awareness

Chapter 7 we discuss modifications to the SPACES model and platform that allow us to register a structuration upon a space in order to support applications and users in engaging with seams. The modifications we propose present the

structuration simply as an annotation to the space which does not affect the calculation of presence and Awareness. A further modification that allows structurations to have an effect on the Awareness reasoning process is another interesting extension to the model to be explored.

Conclusion

Context-Aware applications employ classes of user data that are widely heterogeneous in nature and whose structures are frequently either dynamic, partially unknown or both. Their reliance on sensed physical data also situates them within the same complex and seamful domains as other Pervasive and Mixed-Reality computing systems making it more challenging to analyse their operation at the higher levels of abstraction employed by models and platforms. Our aims in this work were to create a model and platform that support reasoning about a user's context within and across a range of different representations of these context-elements, regardless of whether the user's mobile device possesses the capability to directly assign her a state within them. During a demonstration, our user encountered issues, including a seam, stemming from concerns at multiple levels of abstraction in our model, platform and application. Architectural and philosophical theories about space and technology provided us with new methods of inquiry for approaching and solving these issues in our model and system and for analysing and constructing responses to seams in Pervasive systems in general. Our experience suggests that context can be engaged with and reasoned about using spatial methods, but that the way in which such models are presented to the user must be carefully considered. We find that architectural theories of space and philosophical theories of technological process can ably support the analysis required for such consideration.

Glossary of Terms

- **Element of context/context element** A property of a user's state which forms part of their context in an application but may be represented in many different ways depending on how it is sensed or gathered. For example, location is an element of user context.

- **Context element representation** One of possibly multiple ways of sensing or gathering the data that specifies an element of the user's context. For example, GPS is a representation of user location as a point in a co-ordinate system annotated with various extra values such as altitude and number of satellites visible.
- **Model** Abstraction of a context-element's representation to a form used for reasoning within a context-aware application. For example, a model may abstract distances between GPS points to their Euclidean distance and discard altitude data and number of satellites visible.
- **Dynamic/Mutable** Those representations whose properties may undergo changes during the runtime of an application, altering the set of possible values they can take or relationships between those values.
- **Static** Those whose properties do not undergo changes during an application's runtime.
- **Partially known** Context representations whose structure and extent must be explored and discovered by developers. Frequently because they are derived from infrastructures owned by third parties and not tasked to provide the form of context they are being used for e.g. GSM cell tower IDs whose primary function is not to provide users with a location.
- **Fully known** Context-representations whose possible values and their relationship are entirely known to developers before runtime.
- **Qualitative Spatial Reasoning** Reasoning about how entities are arranged in space in a topological way without referring to numerical coordinates, locations or distances.
- **Point Set** The set of possible all values a user's state could take within a context-representation.
- **Continuous** A context-element model whose Point Set is similar in structure to some subset of $\mathbb{R}^n$.
- **Discrete** A context-element model whose Point Set has properties more similar to a graph-based model.
- **Payload** A bundle of information, services, or behaviours which users gain access to in given contexts.
- **Distance Metric** A mapping on a Point Set that gives the Real and positive distance between any two points in the set (which may also be infinity).
- **Symmetric** A Distance Metric which maps an unordered pair of points to a single distance value, indicating the distance between them is the same regardless of whether it is measured from the first to the second or vice versa.
- **Asymmetric** A Distance Metric which maps an ordered pair of points to a distance value and may map the same pair of points in reversed order to a different value, indicating that the distance between depends on the direction in which it is measured.
- **Area of Interest/Focus** The area a user pays active attention to.
- **Area of Influence/Nimbus** The area a user acts directly upon.
- **Aura** The user's Focus and/or Nimbus.
- **Producer** A user with a Nimbus who wishes to distribute Payloads.
- **Consumer** A user with a Focus who wishes to receive Payloads.
- **Intent** Whether a user is a Producer, a Consumer or Both, and the Payload they distribute if any.

- **Presence** All of a user's locations, Auras, and Intents taken together.
- **Point of Presence** A single user location taken with its Aura and Intent.
- **Awareness** a description and measure of the overlap between the Points of Presence of one user and another.
- **Qualitative Awareness** A description of the topological nature of the overlap between Points of Presence of one user and another classified as None, Low, Passive, Active and High.
- **Quantitative Awareness** A numerical measure of the degree of overlap between the Points of Presence of one user and another.
- **Projection Point** A relationship between two points in different context-element models that indicates they signify the same point in some physical or conceptual aggregate space.
- **Native Representation** A context representation that a user gains presence in by use of sensing or context-gathering technologies.
- **Image/Projected Presence** A Point of Presence created for the user by the model in a non-native representation using its knowledge of the Projection Point relationships between the user's native representation and the non-native one.
- **Aggregated Awareness** When one user has awareness of another in both their native space and that of the other user (via their projected presence in the other user's native space) those awarenesses are combined to give a single overall awareness.
- **Continuity of Context** is when small changes to a user's presence do not result in large changes to her Awareness.
- **Bounding Box** a programmatic construct used where a collection of points is involved in a projection point pairing to wrap and represent that collection. Depending on developer's implementation choices may be a true bounding box or simply a set of points.
- **Intensive History** A form of user history that is cognitively maintained by the user herself. It is rich in her fully lived experience but persists in this richness only for a short period of time
- **Extensive History** A form of user history maintained by the application which is necessarily not so rich as the user's personal experience but is transmissible, durable and can be made available to other users and applications.
- **Survey Awareness** Used for activities where a simple view of the space is required. Defines proximity as being located within a given boundary rather than dependent on the negotiated scale of Nimbus/Focus overlap.
- **Interactional Awareness** The awareness measure that our model initially used as defined in chapters 3 and 4, which is dependent on the negotiated scale of overlap between Consumer Focus and Producer Nimbus. Renamed here to help differentiate it from Survey Awareness.
- **Transduction** A process that occurs across domains, mediums or milieu between which some disparity has arisen and out of which some new activity, process or being generates itself as the result of restructuring one of the involved domains to resolve the disparity.
- **Framing** The presentation of a space, seam or process which brings forward certain of its properties or aspects, conditioning the user's understanding of it and response to it.

References

ABOWD, G. D., et al. (1997). "Cyberguide: A mobile context-aware tour guide." Wireless networks **3**(5): 421-433.

ABOWD, G. D., et al. (1999). Towards a better understanding of context and context-awareness. Handheld and ubiquitous computing: Springer.

AFYOUNI, I., et al. (2012). "Spatial models for context-aware indoor navigation systems: A survey." Journal of Spatial Information Science **1**(4): 85--123.

BECHER, B. and H. BECHER *Six Spherical Gasholders* (1972-1996), *Sonnabend Gallery,* New York, USA http://www.christies.com/lotfinder/AAA/AAA-4709008-details.aspx Accessed: 26th September 2016.

BEHAVIO Google Inc, (2012), Retrieved From: http://funf.org/ Accessed: 6th September 2015, Archived at: http://web.archive.org/web/20150906070906/http://www.funf.org/.

BENFORD, S., et al. (2006). "Can you see me now?" ACM Trans. Comput.-Hum. Interact. **13**(1): 100-133.

BENFORD, S. and L. FAHLÉN (1993). A spatial model of interaction in large virtual environments. Proceedings of the Third European Conference on Computer-Supported Cooperative Work 13–17 September 1993, Milan, Italy ECSCW'93: Springer.

BENJAMIN, D., et al. (2001). New interaction partners for environmental governance : amphibious architecture In Sentient City. M. Shepard (Ed.), MIT Press**:** 48-63.

BENNETT, F., et al. (1997). "Piconet: Embedded mobile networking." Personal Communications, IEEE **4**(5): 8-15.

BOBICK, A., et al. (1996). "The KidsRoom: A perceptually-based interactive and immersive story environment." Presence: Teleoperators and Virtual Environments **8**(4): 367-391.

BOWDEN, S. (2012). Gilles Deleuze, a Reader of Gilbert Simondon. In Gilbert Simondon: Being and Technology. A. De Boever (Ed.), Edinburgh University Press: 135-151.

BROLL, G. and S. BENFORD (2005). Seamful Design for location-based mobile games. In Entertainment Computing-ICEC 2005, Springer: 155-166.

BROLL, G., et al. (2006). Exploiting seams in mobile phone games. Proc. of 3rd International Workshop on Pervasive Gaming Application (PerGames2006).

BROWN, P. J. (1995). "The stick-e document: a framework for creating context-aware applications." ELECTRONIC PUBLISHING **8**: 259-272.

BURKE, J. A., et al. (2006). Participatory sensing. In: Workshop on World-Sensor-Web (WSW'06): Mobile Device Centric Sensor Networks and Applications.

CAKMAKCI, O., et al. (2002). Context awareness in systems with limited resources. Notes of the ECAI-Workshop on Artificial Intelligence in Mobile Systems.

CHABOT, P., et al. (2013). The Philosophy of Simondon: Between Technology and Individuation. London, England: Bloomsbury Academic.

CHALK, W. (1966). Housing as a Consumer Product. Arena. **81:** 228-230.

CHALMERS, M. (2004). Equator: mixing media and showing seams. Proceedings of the 16th conference on Association Francophone d'Interaction Homme-Machine: ACM.

CHALMERS, M. (2004). Space/place reconsidered. Spaces, spatiality and technology conference, Napier University, Edinburgh.

CHALMERS, M. and A. GALANI (2004). Seamful interweaving: heterogeneity in the theory and design of interactive systems. Proceedings of the 5th conference on Designing interactive systems: processes, practices, methods, and techniques: ACM.

CHALMERS, M. and I. MACCOLL (2003). Seamful and seamless design in ubiquitous computing. Workshop At the Crossroads: The Interaction of HCI and Systems Issues in UbiComp.

CHEN, H., et al. (2003). "An ontology for context-aware pervasive computing environments." The Knowledge Engineering Review **18**(03): 197-207.

CHEN, J. (1996) Computational Geometry: Methods and Applications [E-Reader Version], Computer Science Department Texas A&M University Retrieved From: http://www.elib.tic.edu.vn:8080/dspace/handle/123456789/12579 Accessed: 15th September 2015.

COMBES, M. and T. LAMARRE (2013). Gilbert Simondon and the Philosophy of the Transindividual. Cambridge, Massachusetts: MIT Press.

COUCHSURFING Couchsurfing International, Inc., (2015), Retrieved From: https://www.couchsurfing.com/ Accessed: 26th September 2016.

COUPIOUS Coupious LLC, (2011), Retrieved From: http://www.coupious.com Accessed: 28th January 2011, Archived at: http://web.archive.org/web/20110128074656/http://coupious.com/.

CRABTREE, A. and T. RODDEN (2008). "Hybrid ecologies: understanding cooperative interaction in emerging physical-digital environments." Personal and ubiquitous computing **12**(7): 481-493.

CUNLIFFE, A. L. (2002). "Social Poetics as Management Inquiry A Dialogical Approach." Journal of Management Inquiry **11**(2): 128-146.

DELEUZE, G. (1953). How do we recognize structuralism? In Desert islands and other texts. New York, NY, Semiotext(e). **1974:** 170-192.

DERRIDA, J. (1982). Margins of Philosophy. Chicago, IL: University of Chicago Press.

DEY, A. K. (2000) Providing architectural support for building context-aware applications (Doctoral Thesis), Georgia Institute of Technology. Retrieved From: http://www.cc.gatech.edu/fce/ctk/pubs/dey-thesis.pdf Accessed: 26th September 2016.

DEY, A. K. (2001). "Understanding and using context." Personal and ubiquitous computing **5**(1): 4-7.

DEY, A. K., et al. (2001). "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications." Human-computer interaction **16**(2): 97-166.

DIX, A. and G. ABOWD (1996). "Delays and temporal incoherence due to the mediated status-status mappings." Acm Sigchi Bulletin **28**(2): 47-49.

DIX, A., et al. (2006). Intelligent context-sensitive interactions on desktop and the web. Proceedings of the international workshop in conjunction with AVI 2006 on Context in advanced interfaces: ACM.

DIX, A., et al. (2005). Managing Multiple spaces. In Spaces, spatiality and technology, Springer**:** 151-172.

DIX, A., et al. (2010). "Spreading activation over ontology-based resources: from personal context to web scale reasoning." International Journal of Semantic Computing **4**(01): 59-102.

DIX, A., et al. (2000). "Exploiting space and location as a design framework for interactive mobile systems." ACM Transactions on Computer-Human Interaction (TOCHI) **7**(3): 285-321.

DIX, A., et al. (2004). AbsenT Presence. In EQUATOR Record and Reuse workshop.

DIX, A., et al. (2006). Formalising performative interaction. In Interactive Systems. Design, Specification, and Verification, Springer**:** 15-25.

DOURISH, P. (2004). "What we talk about when we talk about context." Personal and ubiquitous computing **8**(1): 19-30.

DROZD, A., et al. (2006). Hitchers: designing for cellular positioning. In UbiComp 2006: Ubiquitous Computing, Springer**:** 279-296.

EFMD *A Magazine Is An iPad That Does Not Work [Video File]* (2011), Retrieved From: https://www.youtube.com/watch?v=aXV-yaFmQNk, Accessed: 26th September 2016.

EGENHOFER, M. J. (1991). Reasoning about binary topological relations. Advances in Spatial Databases: Springer.

ERICSON, C. (2004). Real-Time Collision Detection. San Francisco, CA: Morgan Kaufmann.

FACEBOOK Facebook Inc., (2016), Retrieved From: https://www.facebook.com/ Accessed: 21st September 2016, Archived at: https://web.archive.org/web/20160904130015/https://www.facebook.com/.

FAILS, J. A. (2009) Mobile collaboration for young children: reading and creating stories (Doctoral Thesis), University of Maryland. Retrieved From: http://hdl.handle.net/1903/9633

FLINTHAM, M., et al. (2003). Where on-line meets on the streets: experiences with mobile mixed reality games. Proceedings of the SIGCHI conference on Human factors in computing systems: ACM.

FLUSSER, V. (Trans.) A. Matthews (1999). The Shape of Things. London, UK: Reaktion Books.

FOURSQUARE Foursquare Labs Inc, (2016), Retrieved From: https://foursquare.com/ Accessed: 26th September 2016, Archived at: https://web.archive.org/web/20160925175701/https://foursquare.com/.

FRANKLIN, D. and J. FLASCHBART (1998). All gadget and no representation makes jack a dull environment. Proceedings of the AAAI 1998 Spring Symposium on Intelligent Environments.

FUJINAMI, K., et al. (2004). Take me with you!: a case study of context-aware application integrating cyber and physical spaces. Proceedings of the 2004 ACM symposium on Applied computing: ACM.

FULLER, M. (2001). Boxes towards bananas : dispersal, intelligence and animal structures. In Sentient City. M. Shepard (Ed.), MIT Press**:** 173-181.

GARCÍA, M. A., et al. (1999). Efficient generation of object hierarchies from 3d scenes. Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on: IEEE.

GÄRDENFORS, P. (2004). Conceptual spaces: The geometry of thought. Cambridge, Massachusetts: MIT press.

GELLERSEN, H. W., et al. (2002). "Multi-sensor context-awareness in mobile devices and smart artifacts." Mobile Networks and Applications **7**(5): 341-351.

GLYPHICS Pop Culture Software. LLC, (2015), Retrieved From: https://itunes.apple.com/us/app/glyphics-augmented-reality/id425707629?mt=8 Accessed: 15th April 2016, Archived at: http://web.archive.org/web/20160415072601/https://itunes.apple.com/us/app/glyphics-augmented-reality/id425707629?mt=8.

GOODREADS Goodreads Inc. , (2016), Retrieved From: http://www.goodreads.com Accessed: 27th September 2016, Archived at: http://web.archive.org/web/20160927060717/https://www.goodreads.com/.

GOOGLE NOW Google Inc., (2015), Retrieved From: https://www.google.com/landing/now/ Accessed: 5th September 2015, Archived at: http://web.archive.org/web/20150901075510/http://www.google.com/landing/now .

GREENHALGH, C. (2002). Equip: a software platform for distributed interactive systems. Equator Technical Report 02-002, University of Nottingham, Nottingham.

GREENHALGH, C. and S. BENFORD (1995). MASSIVE: a distributed virtual reality system incorporating spatial trading. Distributed Computing Systems, 1995., Proceedings of the 15th International Conference on: IEEE.

GROSS, T. and W. PRINZ (2003). Awareness in Context. In ECSCW 2003: Proceedings of the Eighth European Conference on Computer Supported Cooperative Work 14–18 September 2003, Helsinki, Finland. K. Kuutti, E. H. Karsten, G. Fitzpatrick, P. Dourish and K. Schmidt (Ed.). Dordrecht, Springer Netherlands**:** 295-314.

GROSZ, E. (2012). Identity and Individuation: Some Feminist Reflections. In Gilbert Simondon: Being and Technology. A. De Boever (Ed.), Edinburgh University Press**:** 37-56.

GROUPON UK My CityDeal Ltd, (2015), Retrieved From: https://www.groupon.co.uk/ Accessed: 26th September 2016, Archived at: https://web.archive.org/web/20160926215404/https://www.groupon.com/.

HARTER, A. and A. HOPPER (1994). "A distributed location system for the active office." Netwrk. Mag. of Global Internetwkg. **8**(1): 62-70.

HARTER, A., et al. (2002). "The anatomy of a context-aware application." Wireless networks **8**(2/3): 187-197.

HILLIER, B. and J. HANSON (1989). The Social Logic of Space. Cambridge, UK: Cambridge University Press.

HOTTOIS, G. (1993). Simondon et la philosophie de la 'culture technique'. Brussels, Belgium: De Boeck Université.

HULL, R., et al. (1997). Towards situated computing. Wearable Computers, 1997. Digest of Papers., First International Symposium on.

IFTTT IFTTT Inc., (2015), Retrieved From: https://ifttt.com/ Accessed: 26th September 2016, Archived at: https://web.archive.org/web/20160926091605/https://ifttt.com/.

IHDE, D. (1986). Experimental Phenomenology: An Introduction. New York: State University of New York Press.

INSTAGRAM Facebook Inc., (2014), Retrieved From: https://www.instagram.com/ Accessed: 14th May 2014, Archived at: https://web.archive.org/web/20150514232614/https://instagram.com/.

ITO, M., et al. (2006). Personal, Portable, Pedestrian: Mobile Phones in Japanese Life. Cambridge, MA: MIT Press.

JEWISH TIME JUMP: NEW YORK Converjent, (2015), Retrieved From: http://www.converjent.org/jewish-time-jump-new-york_page/ Accessed: 12th July 2014, Archived at: http://web.archive.org/web/20140712050448/http://www.converjent.org/jewish-time-jump-new-york_page/.

KHALID, H. and A. DIX (2014). Extended Episodic Experience in Social Mediating Technology: Our Legacy. In Social Computing and Social Media: 6th International Conference, SCSM 2014, Held as Part of HCI International 2014, Heraklion, Crete, Greece, June 22-27, 2014. Proceedings. G. Meiselwitz (Ed.). Cham, Springer International Publishing: 452-461.

KHAN, I. *Every...Bernd And Hilla Becher Spherical Type Gasholders* (2004), *Saatchi Gallery,* London, Uk http://www.saatchigallery.com/artists/artpages/idris_khan_becher_gas.htm Accessed: 26th September 2016.

LAST.FM Last.fm Ltd., (2016), Retrieved From: http://www.last.fm/ Accessed: 27th September 2016, Archived at: http://web.archive.org/web/20160927084003/http://www.last.fm/.

LEFEBVRE, H. (1991). The production of space. Oxford, UK: Oxford Blackwell.

LEONHARDI, A., et al. (1999). Virtual information towers-a metaphor for intuitive, location-aware information access in a mobile environment. Wearable Computers, 1999. Digest of Papers. The Third International Symposium on.

LIU, J. and L. K. DANESHMEND (2004). Spatial reasoning and planning: geometry, mechanism, and motion. Berlin, Germany: Springer Berlin Heidelberg.

LOPES, A. and J. L. FIADEIRO (2005). Context-Awareness in Software Architectures. In Software Architecture: 2nd European Workshop, EWSA 2005, Pisa, Italy, June 13-14, 2005. Proceedings. R. Morrison and F. Oquendo (Ed.). Berlin, Heidelberg, Springer Berlin Heidelberg: 146-161.

MACCOLL, I., et al. (2002). Seamful ubiquity: Beyond seamless integration. Proc. Ubicomp 2002 Workshop on Models and Concepts for Ubiquitous Computing.

MACKENZIE, A. (2002). Transductions: Bodies and Machines at Speed. London, England: Continuum.

MAISONNASSE, J., et al. (2006). Attentional model for perceiving social context in intelligent environments. In Artificial Intelligence Applications and Innovations, Springer: 171-178.

MCCOWAN, I., et al. (2005). Towards computer understanding of human interactions. In Machine Learning for Multimodal Interaction, Springer: 56-75.

MCCULLOUGH, M. (2004). Digital Ground: Architecture, Pervasive Computing, and Environmental Knowing. Cambridge, Massachusetts: MIT Press.

"AOL UNVEILS 'ENTRANCE' ENTERTAINMENT APP FOR WINDOWS PHONE" (2012, 5th September 2012). Mlot, S. Retrieved: 27th September 2016, from http://www.pcmag.com/article2/0,2817,2409292,00.asp.

NELSON, G. J. (1998) Context-aware and location systems (Doctoral Dissertation), Clare College, University of Cambridge. Retrieved From: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.101.8185&rep=rep1&type=pdf Accessed: 26th September 2016.

NEST Nest Labs, Inc. , (2015), Retrieved From: https://nest.com/ Accessed: 2nd September 2015, Archived at: https://web.archive.org/web/20150902235253/https://nest.com/.

NIJHOLT, A., et al. (2009). "Mixed reality participants in smart meeting rooms and smart home environments." Personal and ubiquitous computing **13**(1): 85-94.

PARALLEL KINGDOM PerBlue, (2013), Retrieved From: http://www.parallelkingdom.com/ Accessed: 5th September 2015, Archived at: http://web.archive.org/web/20150905200800/http://www.parallelkingdom.com/.

PATEL, S. N., et al. (2006). Powerline positioning: A practical sub-room-level indoor location system for domestic use. In UbiComp 2006: Ubiquitous Computing, Springer: 441-458.

PEDERSEN, E. R. and T. SOKOLER (1997). AROMA: abstract representation of presence supporting mutual awareness. Proceedings of the ACM SIGCHI Conference on Human factors in computing systems: ACM.

PURBRICK, J. and C. GREENHALGH (2000). Extending locales: awareness management in MASSIVE-3. Virtual Reality, 2000. Proceedings. IEEE.

RANDELL, D. A., et al. (1992). A spatial logic based on regions and connection. KR'92. Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference.

REEVES, S., et al. (2006). The spatial character of sensor technology. Proceedings of the 6th conference on Designing Interactive systems. University Park, PA, USA, ACM: 31-40.

RENZ, J. (2002). Qualitative spatial reasoning with topological information. Berlin, Germany: Springer-Verlag.

RODDEN, T. (1996). Populating the application: a model of awareness for cooperative applications. Proceedings of the 1996 ACM conference on Computer supported cooperative work. Boston, Massachusetts, USA, ACM: 87-96.

RODDEN, T., et al. (1998). Exploiting context in HCI design for mobile systems. Workshop on human computer interaction with mobile devices, Glasgow, Scotland.

ROXIMITY Roximity, Inc., (2015), Retrieved From: http://roximity.com/ Accessed: 29th July 2015, Archived at: https://web.archive.org/web/20150729122222/http://roximity.com/.

RYAN, N. S., et al. (1997). Enhanced reality fieldwork: the context-aware archaeological assistant. Computer applications in archaeology, University of Birmingham: Archeopress.

SADLER, S. (2005). Archigram: architecture without architecture. Cambridge, Massachusetts: Mit Press.

"HOW TO USE AUGMENTED REALITY FOR LEARNING IN CLASSROOM" (2014). Saha, A. Retrieved: 25th September, 2016, from http://www.techgyd.com/use-augmented-

reality-learning-classroom/13780/ archived at http://web.archive.org/web/20150313021349/http://www.techgyd.com/use-augmented-reality-learning-classroom/13780/.

SAUVANARGUES, A. (2012). Crystals and Membranes: Individuation and Temporality. In Gilbert Simondon: Being and Technology. A. De Boever (Ed.), Edinburgh University Press: 57-70.

SCHILIT, B., et al. (1994). Context-aware computing applications. Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on: IEEE.

SCHILIT, B. N. and M. M. THEIMER (1994). "Disseminating active map information to mobile hosts." Netwrk. Mag. of Global Internetwkg. **8**(5): 22-32.

SCHILIT, W. N. (1995) A system architecture for context-aware mobile computing (Doctoral Dissertation), Columbia University. Retrieved From: http://sites.google.com/site/schilit/schilit-thesis.pdf Accessed: 26th September 2016.

SHEPARD, M. (2011). Toward the Sentient City. In Sentient City: Ubiquitous Computing, Architecture, and the Future of Urban Space. M. Shepard (Ed.). Cambridge, Massachusetts, MIT Press: 16-37.

SIMONDON, G. (1992). "The genesis of the individual." Incorporations **6**: 296-319.

SIMONDON, G. (2005). L'individuation à la lumière des notions de forme et d'information. Grenoble, France: Jérôme Millon,.

SIMONDON, G. (2012). Du mode d'existence des objets techniques. Paris, France: Aubier.

SMITH, T. R. and K. K. PARK (1992). "Algebraic approach to spatial reasoning." International Journal of Geographical Information Systems **6**(3): 177-192.

SMITHSON, J. *Rubin's Vase* (2007), Wikimedia Foundation, Inc. Retrieved From: https://commons.wikimedia.org/wiki/File:Rubin2.jpg,

SOHN, T., et al. (2006). Experiences with place lab: an open source toolkit for location-aware computing. Proceedings of the 28th international conference on Software engineering. Shanghai, China, ACM: 462-471.

SPOTIFY Spotify Ltd., (2016), Retrieved From: https://www.spotify.com/uk/ Accessed: 29th July 2016, Archived at: http://web.archive.org/web/20160729065003/https://www.spotify.com/uk/.

STUMBLEUPON StumbleUpon Inc., (2015), Retrieved From: http://www.stumbleupon.com Accessed: 27th Spetember 2015, Archived at: http://web.archive.org/web/20150913152034/https://www.stumbleupon.com/.

TARZIA, S. P. (2011) Acoustic sensing of location and user presence on mobile computers (Doctoral Dissertation), Northwestern University. Retrieved From: https://stevetarzia.com/papers/NWU-EECS-11-09.pdf Accessed: 26th September 2016.

TASKER (4.7) Crafty Apps EU, (2015), Retrieved From: https://play.google.com/store/apps/details?id=net.dinglisch.android.taskerm Accessed: 5th September 2015, Archived at: http://web.archive.org/web/20150905203156/https://play.google.com/store/apps/details?id=net.dinglisch.android.taskerm.

TELLER, S. J. (1992) Visibility computations in densely occluded polyhedral environments (Doctoral Dissertation), University of California at Berkeley. Retrieved From: http://digitalassets.lib.berkeley.edu/techreports/ucb/text/CSD-92-708.pdf Accessed: 26th September 2016.

TUMBLR Yahoo Inc., (2016), Retrieved From: http://www.tumblr.com Accessed: 27th September 2016, Archived at: https://web.archive.org/web/20160927113214/https://www.tumblr.com/.

TVERSKY, B. (1992). "Distortions in cognitive maps." Geoforum **23**(2): 131-138.

TWITTER Twitter Inc., (2016), Retrieved From: https://twitter.com/ Accessed: 26th September 2016, Archived at: https://web.archive.org/web/20160925180149/https://twitter.com/.

ULLMER, B. and H. ISHII (1997). The metaDESK: models and prototypes for tangible user interfaces. Proceedings of the 10th annual ACM symposium on User interface software and technology. Banff, Alberta, Canada, ACM**:** 223-232.

VAN HULLE, M. M. (2012). Self-organizing maps. In Handbook of Natural Computing, Springer**:** 585-622.

VAN LAERHOVEN, K. and O. CAKMAKCI (2000). What shall we teach our pants? Wearable Computers, The Fourth International Symposium on: IEEE.

VOUCHERCLOUD Invitation Digital Ltd, (2015), Retrieved From: https://www.vouchercloud.com/ Accessed: 13th July 2016, Archived at: https://web.archive.org/web/20160713111017/https://www.vouchercloud.com/.

VUKOVIC, M. and P. ROBINSON (2007) Context aware service composition (Technical Report UCAM-CL-TR-700), University of Cambridge. Retrieved From: http://www-ipv4.cl.cam.ac.uk/techreports/UCAM-CL-TR-700.pdf

WALLGRÜN, J. O. (2009). Hierarchical Voronoi Graphs: Spatial Representation and Reasoning for Mobile Robots: Springer-Verlag Berlin Heidelberg.

WANT, R., et al. (1992). "The active badge location system." ACM Transactions on Information Systems (TOIS) **10**(1): 91-102.

WARD, A., et al. (1997). "A new location technique for the active office." Personal Communications, IEEE **4**(5): 42-47.

WEISER, M. (1994). "The world is not a desktop." interactions **1**(1): 7-8.

WITHINGS Withings SAS (2015), Retrieved From: http://www.withings.com/uk/en/ Accessed: 7th September 2015, Archived at: http://web.archive.org/web/20150907170039/http://www2.withings.com/uk/en/?

YANG, J., et al. (1999). Smart sight: a tourist assistant system. Wearable Computers, 1999. Digest of Papers. The Third International Symposium on: IEEE.

ZHENG, V. W., et al. (2010). Collaborative location and activity recommendations with GPS history data. Proceedings of the 19th international conference on World wide web. Raleigh, North Carolina, USA, ACM**:** 1029-1038.