Abstract

Web surfing, the act of following links of interest with no pre-defined search goal, is a paradigm that can be translated to the physical realm of urban exploration. With mobile computing technology and its supporting infrastructure becoming ever more ubiquitous, a user's digital device can be transformed into a portal that connects their physical environment with the virtual, providing instant access to a plethora of information that can influence and guide their interactions with the city. This paper describes the technical aspects of a context-aware system for urban exploration based on the paradigm of web surfing. An implementation is presented that demonstrates a browsing style of interaction with an urban environment through context-based navigational prompts.

1. Introduction

As urban environments become increasingly populated by mobile technologies and their supporting infrastructure, new methods of interacting with our surrounds emerge. Sensors capture environmental cues to trigger context-dependent information and mobile devices store our schedules and preferences, channelling information through increasingly personalised filters. In this third wave of computing it is becoming necessary to control the increasing amounts of oncoming information through a contextual lens, shifting the focus from unlimited access to customised access. This research proposes a framework that supports context-based exploration of urban environments and demonstrates its functionality and informational flow between the elements of the system.

Navigation through both information and urban landscapes is rapidly changing with the proliferation of networked devices in cities. With just a few clicks one can access geocoded data to find the nearest convenience store or even to locate nearby individuals in one’s social circle. With GPS technology new opportunities for location-based entertainment emerge, yet despite an increasing trend towards spontaneity in our communications, navigational systems maintain fixed destination input, with the user entering precise locations. Spontaneity and serendipitous encounters are not encouraged in such systems and there is room for a new tool for mobile exploration that supports and encourages such behaviour. In contrast to directed search, web surfing is a method for users to traverse the net from link to link with no end goal: no real search criteria. The enjoyment of such non-directed “search” arises from the journey itself and from the discovery of unexpected information along the route. This metaphor can be applied to a networked urban environment to change the way interactions within the space occurs.

1 Faculty of Architecture, Design and Planning, The University of Sydney, Australia. {mercedes@arch.usyd.edu.au | marcaurel@usyd.edu.au}
2. Mobile navigation tools

We are accustomed to seeing physical spaces enhanced with localised information. Signposts and street numbers assist in locating ourselves in space and help us conduct physical search. Our physical location is a very powerful indicator of the kinds of information we need access to at any given point in time. Location-based information systems connect items of information to a particular coordinate in physical space. At a later time, users are able to access this information (text, images, URLs, videos) with a mobile device, thus achieving some level of contextual awareness – that of location [2, 5, 10, 16]. As this system becomes more commonplace a further level of context awareness must be implemented to save users from informational overload. This could be achieved by invoking user-identity as a filtering device. Espinoza [5] suggests a method of enhancing access to digital information spaces by filtering information through the matching of user’s history to that of other users. This can be done with a recommender system.

2.1. Recommender systems

Recommender systems identify content relevant to individuals by matching the individual’s profiles with the profiles of a community of users [7]. To illustrate the power of recommender systems, examine Amazon.com’s product recommendation system. When a potential customer views a product, the web page displays other items purchased by customers in conjunction with the queried product. The items displayed are statistically significant, that is, a high number of other customers will have made these purchases together with the queried product. Thus, a correlation is made based on the assumption that customers interested in a particular product share similar tastes and interests. Recommender systems are successfully utilised in online dating services to match users with other users through collaborative filtering algorithms that require users to assign weightings to randomly selected results (profiles of other users) to build up an understanding of their likes and dislikes. This is not always useful as users may be unknown to each other, as in the case of this research. Methods such as those employed by PHOAKS (with anonymous recommendations) or ReferralWeb, (combining links between people to form referral chains) [11], are more useful in the context of this research as they do not require explicit entry, but base their weightings on the information inherent in the data sources.

Recommender systems are economies of scale; the more detailed the information that is provided to the system, and the greater the number of participants, the more accurate the recommendations that can be made. The recommender system is an effective way of filtering relevant information from existing records in a large dataset to make suggestions. It is able to assign levels of similarity between individuals and use the data from individuals to inform other users with close matches, providing a form of context-awareness. The implementation developed here is based on the assumption that a recommender system is used to assign weightings to nodes, but other systems that successfully determine similarity among user profiles may also be valid.

2.2. Context awareness

Context aware computing is a paradigm used in mobile computing research to describe applications that utilise information about the conditions the user is situated in. Schmidt et al. [13] define context as knowledge about the user’s and information technology’s state, including the location, surroundings and situation. Others [12, 15] describe three types of context: (i) computing context, including network connectivity, communication costs, bandwidth and computer hardware, (ii) user
context including the user profile, location, proximity to others and current social situation, and (iii) physical context such as lighting, noise levels, traffic conditions and temperature. Chen and Kotz [3] recommend in addition a time context referring to the time of day, week, month and season and define two types of context; active context, that influences the application, and passive context, that is relevant but not critical to the application. From the specific to the more general, Dey and Abowd [4] describe context as any information that can be used to characterise the situation of an entity, where the 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.

Context is used in this research to support individual requirements. Two types of context are addressed, that of the user’s context and that of the physical context as described by Schilit, Adams and Want [12].

3. Overview and Architecture

This framework is built on the context classification proposed by Dey and Abowd [4] to support a personalised, urban exploration system. Although others [1, 6, 13, 14] have proposed frameworks for context aware computing, their focus has been on the development of systems that can detect and interpret contextual aspects of an environment, emphasising the software engineering aspects of programming sensors to specify contexts or developing rules and architectures to hierarchically sort and classify contexts. Other research [5, 8] has focused on location as the sole input of context, but does not provide the necessary mechanisms to support multiple contextual inputs, which is required for this research.

The aims of the framework can be divided into the following four points:
1. Incorporate a variety of input types, in addition to location
2. Provide a mechanism for sorting input data into contexts
3. Use context information as a filter in a navigational situation
4. Provide the system with feedback to allow the adjustment of context groupings

3.1. Framework information flow

At the topmost level, the framework is comprised of its users, their environment, server-side processes and the device used to facilitate information flow. The context diagram, Figure 1, depicts

```
Environment

User  Client  Server

Environmental information
```

Fig 1. Information flow between elements in the framework.

Individuals in this framework have their own unique context, comprised of their personal details and route history. User inputs take the form of explicitly entered information (i.e. manually entered user profiles), and passively obtained information (i.e. provided by allowing physical location to be tracked).
The user’s context information is used to tag paths, which provides the system with the necessary information to recommend possible paths to new users with similar contexts. Users have a number of roles within the framework, from providing crucial data to allow the system to make recommendations, to utilising the outcomes of the recommendation process to explore the environment. They both define the paths as well as edit their definitions. The client’s main function is to facilitate communication between the user and the remote server and to receive environmental cues. The client will most likely take the form of a device that offers portability, wireless connection to a remote server, the ability to receive environmental information, and contains an interactive information display. Examples of such devices include mobile phones, personal digital assistants, laptops and some digital wristwatches.

3.2. Server information flow

The environment is the physical location users are immersed in and contains the user, the client and the infrastructure required to (i) enable communication between the user and the remote server, and (ii) provide the client with access to environmental information. An example of environmental information is that of the user’s current, physical location, as described by GPS coordinates. The server exists to process the data collected by the client and return queries to the user through the client. It is necessary as a central repository for storing and updating user information and as a place for heavy processing to take place that cannot occur on the client due to limited processing power or space. The server consists of a network interface, database and the recommender system. These elements are shown in Figure 2, within the context of the system initially depicted in Figure 1. The network interface is responsible for communication between the client and the database and between the database and the recommender system. Any other processes that are too processor-intensive for the client are also included on the server.

![Fig 2. Server informational flow.](image)

The arrows in Figures 1 and 2 depict information exchange between the elements within the system. This is a bi-directional flow, with the exception of environmental information, which feeds the client with data such as location information, but does not receive any feedback or outputs from the system in return. Figure 3 depicts informational flow between elements on the server.

3.3. Network interface information flow

The client sends the server user profile and the current GPS coordinates. The database stores user context, profile, node and link information (tagged by the unique identifier of the GPS coordinate). The recommender system finds a context based on the user profile and sends it as a search filter to the database when retrieving nodes. The unweighted nodes are passed to the rule-based weighting algorithm, which assigns them a weighting. The recommendation is made to the client based on the node(s) with the highest weighting(s).
Each user has a history and context that must be created and assigned at the time of user creation. As data is added, it is used to influence which locations (nodes) are presented to the user and other users as possible destinations. User history can be obtained from three pieces of information recorded by the client: the unique user identity, timestamp and GPS coordinates. Using this information, user positions in space and time can be linked to create paths, which represent a journey through the environment. Paths commence at the point where users become visible to the system and end when the user remains at a single node for a prolonged period. They adopt the common contexts associated with the links that join to create the path. Paths tagged by context information can be activated by users with similar contexts on different occasions. The greater the contextual information associated with a path, the more accurate the recommendation can be.

Links require three pieces of information: (i) the unique user identifier, (ii) the start and end node of the link, and (iii) the user’s context as determined from their linked context information. The reason contexts are connected to links is to provide the necessary information to filter and present nodes to other users with similar contexts. Tagging links and therefore nodes with context information is a simple and efficient way to allow the database to be queried for suitable nodes (where suitability is determined by the recommender system). Nodes can be created manually by entering GPS coordinates, or automatically, as defined by the system. Node creation is covered in section 4, where the implementation is discussed.

3.4. Node weight
A weighting is calculated in order to determine which node (and therefore which path) to recommend to the user. Nodes are assigned weightings based on extrinsic values such as their popularity and proximity to other nodes of interest, and intrinsic values, which are taken from user information. Values are decreased if the node appears in the user’s history, designed to reduce the likelihood that paths are retraced. Figure 4 depicts an increase of weighting at node C to take into account the popularity of a nearby node, E. Nodes B and C are equidistant from node A, which is where the user is located. The high weighting of node E influences the weighting of node C, the node the user is ultimately presented with. The weighting of C is the mean between C and E.
It is crucial for the system to be able to adapt to its user and it does this through a negative feedback cycle where feedback on its context-based filtering mechanism is received and used to inform its recommendations, thus influencing the information it suggests to its users. Feedback is provided in two ways, the implicit: by checking to see whether the user has accepted the suggestions made by the system and the explicit: by the user directly changing the information they have control over, such as their profile. The continuous iterations allow for subtle adjustments in the weightings given to nodes and in the way users are compared to others. The system is never static, but continues to adjust and refine based on feedback.

4. Implementation

The implementation was developed with Sun Microsystem’s Java and Java Web Services platform running on Apache Tomcat server. The interface was based on the Google Maps JavaScript API with records stored in a MySQL database. The hardware used in this implementation was an Apple Macintosh PowerBook G4 running Macintosh operating system 10.3.9 with a 1.25 Gigahertz processor and 1 Gigabyte of DDR SDRAM memory. The MySQL database was stored on a remote Server and built and populated using Navicat version 5.3.1. The software used for development was the Java 2 runtime environment, Apache Tomcat server version 4, and the Mozilla Firefox web browser version 2.0.0.4. For the purposes of this implementation, the interface was displayed on a Google Maps Mashup, embedded in an HTML webpage. GPS coordinates were provided by the user's position on the map, which could be changed by scrolling with the arrow keys.

Four aims were defined for the implementation:

1. To demonstrate “browsing”
2. Represent aspects of path creation, selection and usage that can be applied to urban environments in general
3. Incorporate context awareness as a way of providing the user with customised results
4. Be designed in a manner that reflects implementation on a mobile device

4.1. Description of the model

The urban navigation system involves communication between a GPS enabled mobile device and a remote server database over wireless Internet, Figure 5. The mobile device sends the remote server two types of information: user information as determined from the mobile device’s settings and location information in the form of GPS coordinates. This information is passed to a filtering element, the recommender system, which positions the information according to the closest similarity it has to previous entries. The group or ‘context’ it falls into is passed to the SQL database as a search filter, revealing entries with similar contexts. Based on the user information, a weighting is calculated for each path to determine the most suitable path to propose. The mobile device receives the results of this process over the wireless network and displays the navigational information to the user who then
changes their location. This provides the system with a new set of coordinates and knowledge of whether the provided path was acted on or not, based on whether the new coordinates match the proposed ones.

![Fig 5. Information flow between elements in system.](image)

Information flow between the various elements of the system is described in the sequence diagram.

![Fig 6. Sequence diagram showing a basic level of information flow.](image)

The first activity is as the mobile device (also called the client) attempts to initiate connection with the server (via the network interface). Once a connection is established, the user is identified, either by providing a unique identification key that corresponds to an entry in the database, or by creating a new profile that is added to the system. The user's context is partly determined by their profile information, which provides the system with information on which nodes to present and partly by their current location, which feeds into a weighting algorithm that assigns levels of importance to nodes within a certain distance of the user's current location based on a set of rules. Thus there are two stages to the recommendation of nodes: the filtering of nodes based on context information and the ranking of these nodes by propinquity.
4.2. Assigning Weightings to Nodes

When nodes are filtered by context information (by the recommender system) all the nodes within the area are passed to the weighting algorithm that determines which nodes of proximity would be of particular interest to the user. This displays only the heaviest node (or nodes if there are several with the same weighting) for the user to consider. If they choose to visit the node, a new set of weighting is calculated for the rest of the nodes. If a single node was presented and was avoided (i.e. another node not presented was visited) the node is demoted in value. A chess game can be used as an analogy to demonstrate the way the interaction between the system and user progresses. The system presents the user with an option and then waits for the user to act on it. Once the user has made that step, it presents the user with another suggestion. At times it may be useful to present multiple options, to give the user greater choice. This is easily implemented by lowering the threshold to allow the top five weightings to be presented to the user.

Weightings are attributed to user profiles as well as to individual nodes. The context given to the user (based on their profile information) determines which nodes are retrieved from the database. It is only then that nodes are given weightings to determine which location to recommend to the user to visit. Nodes are bounded by their distance from the user’s current position and recalculated each time the user changes location. The nodal weighting is necessary for sorting the recommended nodes by order of appropriateness in relation to other nodes within the selection, as measured by a set of rules.

User profiles are weighed in relation to other users of the system to find matches based on similarity. Each profile is compared to others and given a weighting between zero and one, with one being the most similar and zero the least. For each pair of profiles between the current profile (of the user) and every other profile, attributes are given between the values of zero and one. This weight is then multiplied by a constant value (in this application, the value is set to 50, which is the initial weight of possible nodes the previous user has visited). Visited nodes are linked and placed in the MySQL database ‘links’ table. Links are associated with users and a history is maintained, allowing paths to be reconstructed.

To create a new node, the user must linger in the same location for a minimum of thirty seconds (in a real-world implementation this period may be longer). No records are ever deleted from the database except rejected nodes, which achieve that status by being avoided by the users over multiple recommendations.

4.3. Interface Design

Route recommendations are depicted on a Google Maps Mashup, which provides the infrastructure to allow additional layers of information to be added and incorporated into the user's information display. This provides the functionality to customise the information received by the user through software or database add-ons. Google Maps exhibits key features of user centric design for route instructions [9], including zoom functionality with scrolling, a context view in the corner that may be active or hidden, and the support of multiple views (diagrammatic, satellite, hybrid – see Figure 7). A more complex interface of the device is still under further development. Hereby issues of screen resolution, user experience, and various menu options have to be tested and explored. Figure 7 depicts the first interface of the prototype and allows testing of basic operands and the above described framework.
5. Conclusion

The prototype is intended as a proof of concept for a context-aware urban navigation system and as such uses simulated data and makes assumptions about its environment.

The role of representation has been identified as a key factor in supporting interpretation and the two elements that impact it the most are: (i) the filtering of information to represent and suggest, and (ii) the manner in which the information is depicted. The first can be said to customise, the second to communicate. Customisation involves selecting information from a pool of data based on an awareness of the user’s current context. The individual user’s context data is first identified, then matched against a database of users with a similarity based measure, such as a recommender system, to determine contextually-appropriate suggestions. Feedback is a crucial component of this system, as similarity is an ever changing variable and as such, an iterative process within the system.

Communication is achieved via the most prolific of methods in use for representing urban forms: the map in plan view, allowing users to situate themselves within the context of their surrounds. The Google Maps API has been proposed as an appropriate platform due to its customisability via mashups and simple user interface.

The virtual information, or meta-information about the locale collected and displayed by the mobile device, can be better understood if it is conceived of as an extra dimension to the physical world, intended to provide additional stimulation to inform user’s interactions. The Internet has allowed several novel forms of communication to emerge, revolutionising communication in a way that carrier pigeon, Morse code and telephone lines did in their time. Through the availability of and access to virtual information while on the move, the user is now, more than ever, connected in real-time to the rest of the world. How this ‘always-online’ status affects their activities is a still to be discovered. One welcome side effect of making recommendations based on contextual similarity is that like-minded people are brought together in space, increasing the chance of unexpected interactions between them and for spontaneous events to occur. These interactions occur because of the self-organising nature of the system and enrich the experiences of individuals by giving them opportunities to engage with the city more intensely.
This research has proposed a framework that supports context-based exploration of urban environments and demonstrated its functionality and informational flow between the elements of the system. The need for context-awareness in mobile computing has been recognised and a method of incorporating context information into the system using a recommender system has been explored. The broad nature of context (Schmidt et al. 1999) has been addressed by incorporating both human as well as environmental factors into the system.

**Acknowledgements**

Thanks go to Shea Goyette for his work in implementing the architecture.

**6. References**


