A Location-Aware Mobile Call Handling Assistant

Ibrahim Bokharouss
Dept of Mathematics and Computer Science
Technische Universiteit Eindhoven
5600 MB Eindhoven, The Netherlands
i.bokharouss@student.tue.nl

Wayne Wobcke, Yiu-Wa Chan, Andrew Limaru, Anna Wong
School of Computer Science and Engineering
University of New South Wales
Sydney NSW 2052, Australia
{wobcke|ychan|alimaru|annawong}@cse.unsw.edu.au

Abstract

With the near ubiquity of mobile phones, people are reachable almost anywhere and at any time. At the same time we see an increasing need for people to limit their interactions on the mobile phone, to control (within socially acceptable bounds) when they are reachable and by whom. The user's location plays a significant role herein. Present mobile phones typically provide static means for managing reachability, such as manual profiles. In this paper, we present a location-aware call handling assistant as a dynamic solution. The assistant runs on mobile devices and enables users to manage calls based on their current context (in particular, their location and activity, the date and time, and the caller and caller's group). The system exploits Bluetooth technology for location determination and for user modelling, a new rule-based technique, Prioritized Ripple Down Rules, which gives the user a high level of confidence in the system's behaviour. We discuss the results of a small-scale user study in which the mobile call handling assistant was simulated on a PDA.

1. Motivation

Mobile phones are nearing ubiquity in the general population of the developed world. This means people can reach, and can be reached by, anyone almost anywhere and at any time. This has led to immense changes in the way people manage their time and social interactions with others. The mobile phone means people can work remotely, can be reached in cases of emergency, and in turn, can reach others when they need to. However, this has also led to a blurring of boundaries between different social spheres, e.g. between work and home. We believe people now perceive a need to limit their social interactions on the mobile phone, to preserve some personal time and space when they are not constantly reachable by just anyone. Thus we see a need for people to personalize their mobile phones, automatically being able to accept or reject calls, have the phone ring or not ring (e.g. in public places) or to automatically send a reply (via SMS) when they do not wish to be contacted.

A consequence of these developments is the emergence of systems, which we label "call handling assistants" in the spirit of Maes's personal assistants [9], that allow a user control over how incoming calls are handled, i.e. whether accepted or rejected, whether the phone rings or not, and whether to automatically send an SMS reply (e.g. to communicate the reason of absence to the caller). While most existing call handling systems base their behaviour on the user's contact list and/or an integrated calendar, in this paper we present an approach that also uses automatically updated location information.

There are two reasons for incorporating location awareness into a call handling assistant. First, observation suggests that a large number of call handling decisions are based on location. We typically want a mobile phone not to ring in places like lecture halls, libraries and cinemas, and in other locations such as hospitals, we would like the phone to be switched off entirely. Likewise, an office worker would prefer not to be disturbed while in a meeting room, but would be available to receive calls while in the lunch room. Moreover, recent sociological studies into the use of mobile phones in Japan have identified "place" as significant in regulating users' behaviour. Most prominently, the use of mobile phones for making and receiving calls is prohibited in public places, including on public transport such as trains and buses, while the use of mobile phones for e-mail is socially acceptable in these contexts, Okabe and Ito [10].

Second, we believe people have an expectation that their digital assistants be more intelligent and dynamic, as an answer to the dynamic world in which people are increasingly expected to be flexible. Sudden changes of plans and unforeseen events are everyday occurrences (mobile phones have even increased the use of *ad hoc* plans for interpersonal coordination). As a simple illustration, consider the scenario of a businessman confronted with an unplanned meeting with a client in the boss's office. In a rush, he could forget to switch profiles on his mobile phone. Since the meeting was unplanned, the integrated calendar would not be of any use for filtering calls, yet, if the phone were to ring, it would be an embarrassment for the businessman and an annoyance to the client.

At the same time, there is a growing trend towards the simplification of interaction with devices in order to improve the user's experience, e.g. Philips' Sense and Simplicity campaign [1]. Relieving the user from tedious tasks by adding "intelligence" to the device is a key focus of this paradigm. From this perspective, a location-aware call handling assistant relieves the mobile phone user from having to repeatedly switch manually between different profiles, and also removes the onus of having to enter every meeting or every visit to the cinema in the phone's calendar.

Providing a mobile phone with location-awareness and automatic rules will not suffice to achieve the desired behaviour, as in some cases, the assistant cannot make the right decision based solely on location information. First, a single location does not carry the same meaning to all users, e.g. a restaurant is typically a leisure place for a customer, but a work place for a waiter. A second reason is the existence of exceptions to any rule. A conference room is a place where usually phones should not be allowed to ring, but what if the user is expecting an emergency call from his pregnant wife? The cultural differences amongst mobile phone users, as indicated by Blom, Chipchase and Lehikoinen [3] form the third reason. For example, as noted above, making and receiving phone calls is not allowed in public spaces in Japan, whereas in the Western world it is an all too common occurrence. The incorporation of user modelling with a high level of control into the assistant is an answer to this need for personalization, even allowing the user to adjust the desired behaviour while travelling.

Thus an approach to user modelling allowing both flexibility and accuracy is critical to developing a usable call handling assistant. In previous work on user modelling for communication management, Djian [5] presented a Bayesian network approach for use in telephone and e-mail management assistants. However, a basic problem with this approach is the difficulty of learning an accurate model based on only limited training data (i.e. actual phone calls and the user's behaviour). In addition, while there are similarities between phone call handling and e-mail management, there are also differences, most particularly the limited information about phone calls available to make a decision (with e-mail, the system has access to the content of the message), making accuracy an even greater concern. Hence, we follow our earlier work on e-mail management, Ho, Wobcke and Compton [7] in taking a rule-based approach to user modelling, in particular based on Ripple Down Rules, Compton and Jansen [4]. Ripple Down Rule systems are suitable for personal assistants because they allow the incremental construction of a user model, are comprehensible by the user, and provide a high level of accuracy.

In this paper, we begin with a discussion of "location" from the user's perspective in relation to the problem of call handling, then present our technologies for implementing a location-aware call handling assistant, including a Prioritized Ripple Down Rules technique for user modelling and a localization method based on Bluetooth beacons. We then present the user interface of the call handling assistant and illustrate its behaviour through a scenario. Finally, we describe the results of a small-scale study to evaluate the usability of the assistant and particularly its rule system.

2. Location Awareness

The call handling assistant must allow the user control over how incoming calls are handled based on the user's location (amongst other factors such as the caller, the caller's group, the current date and time and the user's current activity). The way locations are conceived and represented thus determines to a large extent the practicality of this feature of the assistant. In fact, the notion of "location" is not a straightforward one, and in this section we discuss various representations for location, motivating our choice of a more complex hierarchical representation mixing physical locations and location types.

A straightforward approach to location is to have the system work with physical locations (e.g. 'Library, University of New South Wales, Sydney' or 'Ritz Cinema, Randwick, Sydney'). In this setting, the user can define rules for locations that relate to daily life: the office at work, sport at school, regularly visited restaurants, etc. Although this approach corresponds to the natural way we speak of locations, its main disadvantage is that it is inflexible and so may lead to a large rule set, since the user has to define a separate rule for every instance of a particular location type. For example, if a user prefers not to be disturbed in restaurants, a rule needs to be defined for every restaurant visited. This may eventually become too complex to manage.

An alternative approach is to represent locations by their logical type (e.g. 'library', 'restaurant' or 'office'). Such a conceptualization addresses the main drawback of using physical locations. The rule set will be minimized drastically, as several different physical locations can be grouped into a single location type. In the above example, this would enable one rule to apply for all restaurants, including those never before visited. There is one drawback of this scheme however. In a case in which the user would like to allow the mobile phone to ring in his own office, but not in someone else's office, the assistant would need a way to differentiate the two physical locations which are both logically seen as an office.

Thus the approach we propose is a more complex hybrid approach, combining both physical locations and location types in a single framework. With a user model based on Prioritized Ripple Down Rules, it is natural for the user to define a generally applicable rule (e.g. one applicable in an office) and also to define an exception to such a general rule (e.g. to apply in his or her own office). Moreover, we allow physical locations to be hierarchically structured (e.g. suburbs within cities, or rooms within buildings). The hierarchical structure of locations allows the user to choose the appropriate degree of granularity for the rules. This enables a user to have very specific rules for highly specific locations (e.g. work, hospital or school) that are frequently visited, while more coarse-grained rules for less frequently visited locations (e.g. other cities). An illustration of a possible location hierarchy is given in Figure 1.

This approach also enables the logical type of a physical location to be time dependent. Consider a location that is used as a meeting room in the morning and as a lunch room in the afternoon. Instead of asking the user to specify separate rules for these different cases, we propose to make the locations programmable. In the morning it would present itself as a meeting room, and in the afternoon as a lunch room. In addition, in the call handling assistant we allow the user to select their location from a drop-down list, overriding the automatically defined location. This might be desirable for two reasons, first to correct anomalies in the location system (e.g. when the location presents itself as one type of location when from the user's point of view it is another), and second, when the user wishes to pretend they are in a location different from their actual location.

+ **Sydney**
    + City Centre
    – Randwick
        – Ritz Cinema
        – Chao Praya Restaurant
    – University of New South Wales
        – Library
        – Central Lecture Theatre Block
        – Computer Science and Engineering
            – Office Room 4.29
        – ...
    + Kings Cross
+ **Melbourne**
    + Fitzroy
    + St Kilda
+ **Perth**
    + ...

Figure 1. Hierarchy of Locations

3. Technologies

3.1. Prioritized Ripple Down Rules

The user model in the call handling assistant is based on Ripple Down Rules. Ripple Down Rules (RDR), Compton and Jansen [4], are systems of 'if-then' rules organized in a hierarchy of rules and exceptions (exceptions to rules may themselves have exceptions, etc.). RDR systems are incremental; starting with a rule base containing only a default rule at the root of the tree, the user repeatedly adds or refines rules (creates exceptions to existing rules) to correctly classify a sequence of examples. This process results in a high degree of accuracy since, whenever the rule base is modified, the RDR engine ensures that all previous examples continue to be classified correctly.

There are various types of RDR system; with Single Classification RDR (SCRDR), the rule base defines one conclusion for any given example, whereas with Multiple Classification RDR (MCRDR), Kang, Compton and Preston [8], there is a set of such conclusions. Each conclusion for an example is found by following a path in the RDR tree to the most specific node whose rule is applicable to the example (i.e. the example satisfies the rule's conditions) and returning the rule's conclusion.

An example MCRDR rule base for the call handling domain is shown in Figure 2 (in section 4, we describe a scenario showing how this rule tree is constructed using the call handling assistant). In this tree, the rules are named *Rule 0*, *Rule 1*, etc., in the order of their creation. The conditions of the rules are constraints on the caller and call group, the time and date, and the user's current activity and location; the conclusion is a "profile" to be used in handling the call, specifying whether to accept or reject the call, and if the call is accepted, whether to allow the phone to ring or not (silent mode), and whether to automatically send an SMS reply or not. A rule matches the current location if the location condition is a substring of the current location as determined from the Bluetooth beacon (see below). The default rule, *Rule 0*, is that any condition gives the *Default* profile, which is under the control of the user. The four rules underneath *Rule 0* are rules defining what to do in various situations, e.g. *Rule 4* states that if the user's location is *Lecture hall* to use the *Busy* profile. *Rule 5* is an exception to this rule; if someone from the *Friends* group calls while in a lecture hall, use the *Unavailable* profile instead. So, for example, if someone from the school calls while the user is in the office, both *Rule 1* and *Rule 3* are most specific rules that apply to this example, and the MCRDR rule base returns the set of conclusions $\{$*Emergency*, *Busy*$\}$.

A key feature of standard RDR systems is that the system provides extensive help to the user in building the rule base, so much so that the user never has to examine the rules themselves in order to resolve conflicts. Rather, a standard

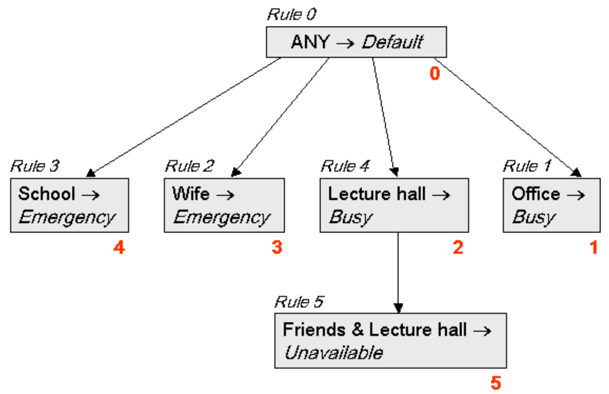


Figure 2. Prioritized Ripple Down Rule Base

RDR system makes use of "cornerstone cases", those examples that triggered the formation of a rule, in order to help the user resolve potential conflicts as new rules are defined. When a new rule is being defined, in the context of a new example, the RDR system presents those cornerstone cases that would conflict with the new rule, allowing the user to adjust the rule's conditions, if necessary, *before* committing to the new rule. In addition, when defining a new rule to correct an incorrect conclusion, the system is able to add all necessary refinements to the rule base at once, greatly simplifying the construction of a complex rule base.

However, in the call handling application, a more "rule-oriented" approach is needed, since users do not deal with sets of examples (calls), though they can use the call history as a repository of examples. Moreover, as the system must produce determinate behaviour for each call, a single classification is needed, suggesting the use of Single Classification RDR. But the rule systems for SCRDR typically become more complex than those for MCRDR (they effectively have only binary branching), so for the call handling assistant, we have chosen to use Multiple Classification RDR with a prioritization mechanism to ensure that only one classification is given for any example (that of the rule with highest priority applying to the example). The priorities define a total ordering on the rules in the RDR tree, and are shown in Figure 2 below each node. There is only one constraint on the priority ordering: any refinement rule must have a higher priority than its parent (so the default rule has the lowest priority of all); otherwise, the priority ordering is defined by the user, and the assistant must provide an intuitive interface for manipulating priorities as well as for defining rules. With the priorities given in Figure 2, if someone from the school calls while the user is in the office, *Rule 3* alone will be used, so the *Emergency* profile will be used in handling the call.

3.2. Bluetooth Localization

To provide the assistant with location awareness, a localization mechanism for the mobile devices must be designed and implemented. Different technologies have been considered, the main concern being the applicability to the call handling domain, in particular being able to model the hierarchical structure of physical and logical locations. Moreover, the type of locations the call handling assistant works with are typically indoors. This requires the technology to facilitate indoor localization on a room level.

GPS (Global Positioning System) is a well developed and widely used technology, but it does not meet the requirements for our domain. GPS does not work indoors, particularly in steel-framed buildings, and its resolution of several metres does not allow it to work with locations on a room level. Moreover, GPS imposes cost and energy consumption requirements that make it unsuitable for deployment on mobile devices.

An approach that is receiving more support in the area of ubiquitous computing is indoor location determination based on wireless networks, in particular Bluetooth [12, 2, 11, 6]. The key advantages of this scheme are the ease of deployment, the wide availability of Bluetooth devices, the low energy consumption and the low cost. This makes it highly suitable for mobile devices.

The backbone of our system is an infrastructure of wireless beacons. Each participating location is equipped with such a beacon, which is programmed to represent its location according to a predefined standard. A beacon remains passive until it receives a request. The mobile device runs the localization software, which is programmed to scan for beacons in range. Upon detection of a beacon, it requests the location from the beacon, and updates the call handling assistant accordingly. No data is sent from the mobile device to the beacon, ensuring the user's privacy. In the case that there is no beacon in range, the location on the assistant is set to *Unknown*.

Instead of using dedicated hardware for the beacons, we chose to use simple Bluetooth dongles connected to desktop computers. Software on the mobile device is programmed to perform a Bluetooth discovery periodically. The discovery sends a request to all Bluetooth devices in range to identify themselves. A list of devices in range, represented by their Bluetooth address and the device name, is returned.

For location setting, we decided not to use look-up tables, but to use the dongle's Bluetooth device name: the name of the location is stored as a string for the device name (to exclude other Bluetooth devices, the names are prefixed with a special sequence *LA*). An example of such a string is *LA*Sydney > University of New South Wales > Computer Science and Engineering > Room 4.29 :: Office. The main motivation for this approach, next to simplicity, is flexibility, i.e. it allows transparent addition and update of locations. Moreover, the network communication is reduced, since the device name is returned by default upon a discovery. This scheme also facilitates the concept of time dependent locations (as discussed in section 2), as a beacon can be easily programmed to change its device name. Finally, as both the physical hierarchy and logical type of the location are stored in the beacon, it removes the need for hybrid methods, such as a beacon network together with GPS.

A major issue for deployment of such a system is the positioning of the beacons. There is a trade off between reliability and resource consumption, i.e. having more beacons allows more accurate localization but is more costly. Our initial design was to have the beacons focused on the entrances to locations. This corresponded to the intuition of entering a location. One advantage of this scheme is that the mobile device does not need to keep polling for beacons once it is inside a room. However, this scheme is not very reliable, e.g. the device might not detect the beacon in the case of a bad network connection, or if the phone is turned off on entrance to a location but switched on inside. Our second approach was to have the beacons cover the entire area. Large rooms therefore need multiple beacons (all configured the same). In this setting, the mobile phone keeps polling periodically, resulting in more traffic but increased reliability.

The main problem caused by this localization scheme is interference between the beacons. The Bluetooth radio signal penetrates constructions such as walls, windows and doors. This means that beacons from multiple rooms could be detected at the same time; for example, if the user is walking along a corridor, the device may receive signals from many of the rooms. Several alternatives have been suggested. Feldmann *et al.* [6] use the radio signal strength (RSSI value) together with the triangulation method to calculate an exact position. Although an improvement, experiments showed that it realizes a position estimation with a precision (RMS) of 2.08 metres. Bandara *et al.* [2] proposed a novel Bluetooth access point that supports variable attenuators to overcome this problem, yet the precision obtained was 92% within 2 metres accuracy. Novel approaches are therefore needed for more accurate location determination. Until then, the most desirable solution is to position the beacons to minimize interference, and we assume this is possible with an acceptable level of reliability.

4. Call Handling Assistant

4.1. Prototype

In this section, we describe a working prototype of a location-aware call handling assistant. The prototype has been implemented as an extension to the rule-based call handling assistant (without location awareness) presented in Wobcke, Chan and Limaru [13]. This system was implemented in Java (Jeode Personal Java 1.1.8) on a PDA (HP iPaq hx4700) using hsqldb as the database engine. The addition of location awareness to the system has been realized by programming the client using JSR82, the Java APIs for Bluetooth (Avetana's Bluetooth stack was used), whereas Bluetooth access points were connected to computers in different rooms to simulate the presence of location beacons.

The system provides the basic functions supplied with a mobile phone, though cannot be used to make calls (calls are simulated through a wireless network). While it would have been desirable to implement the assistant on a mobile phone platform, at the time of development, the hx4700 had a faster processor, more memory and a larger (4 inch) screen with four times the resolution of the closest comparable mobile phone. This enabled the system to be evaluated in a user study now with a view to future deployment on more powerful mobile phones.

The graphical user interface is organized as a series of tabs: Rules, Profiles, Phonebook, Calendar and History, as shown in Figure 3(a). The Phonebook allows the user to set up a list of contacts and call groups (with individuals possibly belonging to multiple groups, as is natural with Multiple Classification RDR). The Profiles tab enables the user to define a range of actions to be taken when receiving a call (accept/reject, ring/silent and send SMS), and there is a *Default* profile which the user can modify. The Calendar tab allows the user to define a list of activities with associated days and times, and the History tab stores a list of calls received. The Rules tab is the most important, as it provides both a mechanism for manipulating rules (adding, refining and prioritizing), and for searching (displaying a subset of the rules that match a particular set of conditions). The search function was anticipated to be useful when users had a large enough rule set, e.g. that would not fit on one screen (around 9–10 rules). Furthermore, the subset of rules resulting from a search can be prioritized with respect to one another and the system will maintain a single consistent priority ordering. This provides a convenient way to prioritize a larger rule set by focusing on those rules that could potentially apply in a given situation or range of situations.

The current mode of the assistant is displayed in the main (blue) title bar, and for Rules is generally one of *search* or *edit*. The default search mode is to display those rules matching the user's current location and the current day and time. These are the current active rules, which are the rules that are applicable in the current situation, giving the user some idea of the range of possible behaviour of the assistant at that time. The rules are displayed in priority order (highest priority at the top). Note, however, that the hierarchical structure of the MCRDR rule base cannot be displayed using this method, and users need a separate mechanism to infer this structure (an error message is displayed when they attempt to redefine the priority of a rule to be lower than that of its parent in the rule tree). When a new rule is added, the system displays the set of rules that are compatible with the new rule (similar to how potential conflicts are shown in standard RDR), being the set of those rules that can be meaningfully prioritized with respect to the new rule. The heading above the list of rules changes to indicate when prioritization of rules is possible. Either the user's activity and the current time, or the current location, are displayed in the (pale blue) bar at the top of the screen below the main title bar. The activity is taken from the user's calendar, though it can also be manually adjusted. The location is obtained from the beacons in range of the mobile phone, or if none is present, is set to *Unknown*. The user can select from a list of locations to override the automatic location setting.

4.2. Scenario

The following scenario illustrates the main features of the call handling assistant, including the use of the Prioritized Ripple Down Rule base for representing the user model and the use of location awareness for dynamically altering the behaviour of the system.

Assume the user has already set up a profile *Emergency* (accept the call and ring), a profile *Busy* (accept the call in silent mode), and the following rules: if location is *Office*, use the *Busy* profile; if caller is *Wife*, use the *Emergency* profile; if caller is *School*, use the *Emergency* profile. As the user enters his office, the system learns via the beacon inside the room about the change of location, and updates the location displayed in the upper right corner. It also automatically shows the rules that apply at the current time in this location, in priority order, as in Figure 3(a).

Later on that day, the user goes to a lecture hall. As there are no rules defined for *Lecture hall*, the assistant notifies the user upon the detection of this new location, and the user can now define rules that apply in this new situation (and later select this location from the drop-down list). Suppose the user adds a new rule: if location is *Lecture hall*, use the *Busy* profile. The assistant displays all compatible rules, and the user prioritizes the new rule, using the large (green) arrow buttons, to be just above the default rule, which always has lowest priority, as shown in Figure 3(b). This means that should the user's wife or the school call whilst the user is in the lecture hall, the phone will ring, as these rules are of higher priority and apply in any location.

The user now wishes to change the behaviour of the system by defining an exception to the 'Lecture hall' rule that applies when the caller is from the *Friends* group. This is done by selecting the rule and pressing the *refine* (R) button. The rule's existing conditions are greyed out (these cannot be modified) and the user selects the *Friends* group and defines a new *Unavailable* profile to reject the call, as in the *Busy* profile, but also to inform the caller via SMS that the user is not free to talk at that time. The user then checks the rules applying in the current location, and prioritizes the new rule above the 'School' and 'Wife' rules, as shown in Figure 3(c). Note that these rules can overlap, so the ordering is significant. For example, if someone from the school happens also to be in the *Friends* group, when they call, they receive an SMS in reply via the *Unavailable* profile, rather than having their call accepted. On returning to the office, the system automatically displays the rules applicable in that location, similar to the display in Figure 3(a).

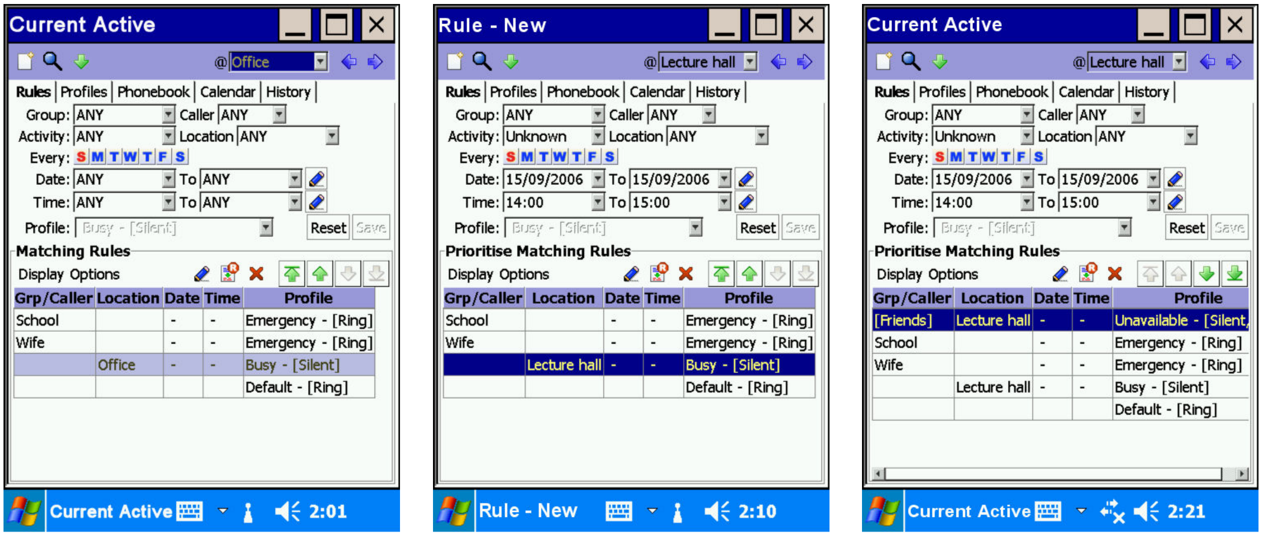
5. User Study

To evaluate the call handling assistant, a small-scale usability study was undertaken. The main objective of the user study was to assess the Prioritized Ripple Down Rules technique, and more particularly the language for expressing rules, for their applicability and usefulness in a call handling application for mobile devices. The second objective was to identify usability issues with the current graphical interface. The third objective was to determine whether users found the concept of a call handling system desirable.

There were five participants in the study, two male and three female, aged from 20 to late-40s, and with a variety of occupations (two students, two in management and one in administration) and level of computer expertise (two with formal computing backgrounds, three without). Each user was given a PDA with the assistant installed, preset with some contacts from their address book obtained previously. Each user interacted with the system in a laboratory setting for three one-hour periods over consecutive days, observed by a researcher who provided minimal help using the system and who simulated calls (over a wireless network) from a number of the user's contacts using a second PDA.

Each test subject was given minimal instructions in very broad terms on the basic purpose of the system: that it could be used to define rules to control the device's behaviour for call handling, and that these rules could have exceptions and be priority ordered. They were asked to consider some situations where special call handling would be desirable, and to define some rules that would be applicable in those situations. They were given minimal instruction about the organization of the interface and the meaning of the icons.

To evaluate the general idea of a call handling assistant, a series of questions were asked after the user study. The results, given in Table 1, where the answers are on a five-point Likert scale with 5 the highest rating, show that the type of assistant envisaged would be highly desirable. The main questions of interest to us, in addition to whether a rule system was useful, were whether users thought the rule system needed exceptions and prioritization. In both cases, users said they would like these features.



(a) (b) (c)

Figure 3. PDA Displays for Sample Scenario

`Would you have liked a system that ...`

Let you organize users into multiple groups	4.6
Let you put users into subgroups	4.6
Let you define rules for activities, ...	4.4
Let you search for specific rules	4.8
Let you make exceptions to a rule	5.0
Let you make some rules more important ...	4.8
Let you see all the rules	5.0

Table 1. Call Handling Assistant Concept

`It was easy to ...`

Change my set of rules	2.8
Change the importance of a rule	2.8
Add a rule	3.6
Define an exception to a rule	3.4
Define how I wanted the system to behave	2.8
Navigate through the system	2.8
Understand the system's feedback	2.6
Recover from any mistakes	2.6
Use the system	2.6

Table 2. Call Handling Assistant Usability

Of particular interest were the questions relating to the usability of the call handling assistant. Again a series of questions were asked after the three testing sessions, with responses on a five-point Likert scale with 5 being 'strongly agree' and 1 being 'strongly disagree'. The results are shown in Table 2. The questions of most interest were whether it was easy to add a rule (3.6), define an exception to a rule (3.4), and change the importance of a rule (2.8), which were rated well. Also of interest was the degree of difficulty users faced in conceptualizing and constructing rule systems; the question on 'defining how I wanted the system to behave' was intended to capture this, and the moderate result (2.8) suggested users had some difficulty.

From Table 2, it can be seen that adding, prioritizing and changing rules were all rated as easy in accordance with the users' behaviour. It is interesting to note that although users rated defining an exception to a rule as being relatively easy, it turned out that users seldom defined refinements to rules using the *refine* button (this information was found from inspection of the system logs). Rather, users tended to define new rules that were specializations of existing rules, which they then regarded as exceptions. In effect, the prioritization mechanism was used to handle exceptions. The rating indicates that users were satisfied with this alternative structure. Why users seldom used the *refine* button was due to two factors. First, users were not told about the *refine* button and may have had difficulty finding it due to its size and location on the screen. Second, the rules users defined were fairly simple and an exception could easily be treated as a new rule with a higher priority. This may increase the flexibility of the system, making it suitable for users who do not require a high level of functionality.

The problematic aspects of the interface were able to be determined from qualitative feedback given during testing rather than through the above questions. First, users expected to be adding a rule when starting the system for the very first time, yet were confused when the default was to show the current rules. Second, navigability was rated as moderate, but observation suggested that users did navigate successfully through the tab menu style organization. Third, at times it was unclear when the system was in edit mode, causing confusion for users and making them more prone to error. What also caused problems was the inability to switch briefly to a different tab (e.g. to look up a profile name) in the middle of adding a rule, without losing information in the initial tab. Some frustration was also caused by a delay in the system's responses (due primarily to the slow speed of the Java GUI on the PDA platform). This caused problems when users, thinking they had not pressed a button, repeatedly pressed the same button, resulting in a chain of events for Java to process. These platform issues can be alleviated with a more efficient implementation.

In summary, users were enthusiastic about the concept of the call handling assistant, generally thought the navigation in the user interface was good and liked the flexibility of the rule definition language. However, the feedback provided in the current system could be improved (especially decreasing the response lag and providing more meaningful error messages), the workflow was not always obvious (the default modes could be more intuitive and changes to the rules displayed as a new rule was added, to show the matching rules, sometimes caused confusion). Finally, though users liked the idea of the call handling assistant and found it relatively easy to manipulate the rule system, it was difficult to assess the effectiveness of the Prioritized Ripple Down Rules technique independent of the interface for rule definition, especially as users defined comparatively few rules.

6. Conclusion

With the prevalence of mobile phones, there is a need for people to more effectively control the nature of their social interactions. In this paper, we presented a location-aware call handling assistant for mobile devices, focusing on the issues of user modelling via Prioritized Ripple Down Rules and localization using Bluetooth beacons. We have shown how location awareness enhances call handling assistants by providing a more expressive and flexible rule language. Prioritized Ripple Down Rules are suitable for incremental construction of a user model giving a high degree of accuracy and comprehensibility, while Bluetooth beacons are especially suitable for indoor localization. The results of a user study support the concept of the call handling assistant, the effectiveness of navigation within the application and the ease of use of the interface for manipulating rules.

Acknowledgements

This work was funded by the CRC for Smart Internet Technology and was conducted while the first author was visiting the University of New South Wales.

References

[1] E. Aarts. Ambient Intelligence Drives Open Innovation. *interactions*, 12(4):66–68, 2005.

[2] U. Bandara, M. Hasegawa, M. Inoue, H. Morikawa and T. Aoyama. Design and Implementation of a Bluetooth Signal Strength Based Location Sensing System. In *Proceedings of the 2004 IEEE Radio and Wireless Conference*, pages 319–322, 2004.

[3] J. Blom, J. Chipchase and J. Lehikoinen. Contextual and Cultural Challenges for User Mobility Research. *Communications of the ACM*, 48(7):37–41, 2005.

[4] P. J. Compton and R. Jansen. A Philosophical Basis for Knowledge Acquisition. *Knowledge Acquisition*, 2:241–257, 1990.

[5] D. Djian. Communication Management: E-Mail and Telephone Assistants. In B. Azvine, N. Azarmi and D. D. Nauck, editors, *Intelligent Systems and Soft Computing*. Springer-Verlag, Berlin, 2000.

[6] S. Feldmann, K. Kyamakya, A. Zapater and Z. Lue. An Indoor Bluetooth-Based Positioning System: Concept, Implementation and Experimental Evaluation. In *Proceedings of the 2003 International Conference on Wireless Networks*, pages 109–113, 2003.

[7] V. H. Ho, W. R. Wobcke and P. J. Compton. EMMA: An E-Mail Management Assistant. In *Proceedings of the 2003 IEEE/WIC International Conference on Intelligent Agent Technology*, pages 67–74, 2003.

[8] B.-H. Kang, P. J. Compton and P. Preston. Multiple Classification Ripple Down Rules: Evaluation and Possibilities. In *Proceedings of the* 9^{th} *AAAI-Sponsored Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, pages 17.1–17.20, 1995.

[9] P. Maes. Agents that Reduce Work and Information Overload. *Communications of the ACM*, 37(7):31–40, 1994.

[10] D. Okabe and M. Ito. *Keitai* in Public Transportation. In M. Ito, D. Okabe and M. Matsuda, editors, *Personal, Portable, Pedestrian*. MIT Press, Cambridge, MA, 2005.

[11] A. Peddemors, M. Lankhorst and J. de Heer. Presence, Location, and Instant Messaging in a Context-Aware Application Framework. In M.-S. Chen, P. Chrysanthis, M. Sloman and A. Zaslavsky, editors, *Mobile Data Management*. Springer-Verlag, Berlin, 2003.

[12] S. Thongthammachart and H. Olesen. Bluetooth Enables In-door Mobile Location Services. In *Proceedings of the 57th IEEE Semiannual Conference on Vehicular Technology*, pages 2023–2027, 2003.

[13] W. R. Wobcke, Y.-W. R. Chan and Y. A. Limaru. A Call Handling Assistant for Mobile Devices. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 717–720, 2006.