Learning User Preferences in a System Combining Pervasive Behaviour and Social Networking

E. Papadopoulou, N.K. Taylor and M.H. Williams
School of Mathematical & Computer Sciences
Heriot-Watt University
Edinburgh, UK
{E.Papadopoulou, N.K.Taylor, M.H.Williams}@hw.ac.uk

S.Gallacher
Intel Collaborative Research Institute
University College London
London, UK
s.gallacher@ucl.ac.uk

Abstract— **Pervasive computing is an important paradigm for interacting with the increasingly complex environment surrounding the user while social networking is equally important for handling the daily interactions between users. One challenge that has been identified recently is to bring these two paradigms together in an integrated way so that the user has the advantages of both, together with benefits arising from the combination of the two. This is the aim of the SOCIETIES project which is developing a system that combines general pervasive system behaviour with social networking in a seamless fashion. One of the important features on which the system is based, is that of context aware personalization, and major components in the SOCIETIES system include those of personalization and context management. Within this, one of the major challenges is that of building up a set of user preferences to obtain the best predictions, especially since users will change their minds about preferences from time to time. To achieve this two different approaches (rule-based and neural net) are used by the system to represent user preferences. This paper describes the smart space approach developed in the Persist project and currently being expanded in SOCIETIES, and then focuses on the problem of managing user preferences. The rule-based approach is described in detail and the neural net approach only briefly. The strategy used employs both approaches and compares the predictions of the two to obtain the best accuracy and adaptability.**

Keywords—pervasive systems; social networking; user preferences; learning; personalization

I. INTRODUCTION

The notion of a social network has been around since the 1800s with significant developments in the first half of last century. However, social networking has been transformed in the past two decades through the emergence of a variety of software systems enabling instant communication with any number of users throughout the world. The combination of the very large scale of operation coupled with ease of use of such systems has led to the rapid adoption and huge success of systems such as Facebook, Youtube, LinkedIn, Flickr, etc. Not only is the number of users using such systems very large but also the time spent on them represents a significant proportion of the time that the average user spends on the computer.

On the other hand, pervasive computing has developed more slowly, facing significant challenges. The need for pervasive computing arises from the increasingly complex environment surrounding the user. As the technology for sensors and other devices has developed, so their cost has fallen dramatically and the range of different applications encompassed by them has grown. The consequence of this has been a steady growth in the number of devices of different kinds (including sensors, computers and general appliances) in the environment, creating an increasingly complex environment surrounding the user, and hence a growing need to provide support to enable the user to control this. This is one of the driving motivations behind the development of pervasive systems [1] – i.e. to provide the support necessary to enable the user to control and manage the growing numbers of devices, networks and services that are available at any time or place. This has led to an increasing amount of research aimed at finding solutions to the problems of pervasive and ubiquitous computing, and more and more prototypes are emerging to test different subsets of ideas in this area.

Although these two paradigms (social networking and pervasive computing) are very different, the one concerned with interaction with other users, the other concerned with interaction with the environment, the two complement each other and we believe that they can be brought together and integrated seamlessly into a single system with the benefits of both – a Pervasive Social Networking (PSN) system. There are potentially significant benefits that could be gained from such a merger. In fact there has been growing interest in extending social networking by combining it with location awareness and there are already a number of applications in which this has been done – for example, systems such as FourSquare rely entirely on this. However, combining social networking with full pervasive system behaviour goes much further than this.

The SOCIETIES project is a large European research project with fifteen partners which is building on recent technical developments in pervasive computing and social networking to create such systems. Development of a basic prototype is almost complete and the process of testing it with the aid of real users is under way.

This paper is concerned with the handling of personalization in such a system and, in particular, the problem of building up a set of user preferences despite changing demands by the user. It describes the approach adopted within the SOCIETIES project to handle this.

978-1-4799-1092-2/13/$31.00 ©2013 IEEE

The following section provides a brief description of the problem being addressed. Section III describes some background to the research while section IV describes the notion of Personal Smart Spaces and their relevance to this work. Section V outlines the relationship between Personal Smart Spaces and fixed smart spaces, and the benefits of using the PSS approach. Section VI provides a brief outline of the process of personalizing the user's physical environment while section VII describes the rule-based approach in more detail. Section VIII outlines the neural net approach very briefly while section IX discusses the combined approach. Section X summarizes and concludes.

II. THE CHALLENGE

The major challenge facing the SOCIETIES project is to combine the functionality of social networking with that of pervasive systems in an integrated way. The approach chosen is based on the use of smart phones as the main user device which can connect to other devices in the user's environment as required. The work has focused on the use of Android devices coupled to the main processing and storage facilities in the cloud.

However, one of the important problems facing such a system, and indeed many other systems in which "smart" technology is used, is the ability to adapt the behaviour provided to take account of the needs and preferences of the individual user depending on the circumstances prevailing at any point in time – or, in other words, it must be both personalizable and context aware. This applies not only to the adaptation of the content provided by services and the manner in which this is presented to the user but also to any actions that the system may perform on behalf of the user. In order to be able to do this, the system must build up sufficient knowledge about the individual user and the adaptations or actions that he/she performs and the context in which these occur. Since it is unrealistic to rely on the user to provide such information directly, the approach generally used is to monitor the user's actions and the context in which they are performed and apply machine learning techniques to infer preferences from this data.

Thus a major challenge facing the developers of pervasive systems lies in how to use the data obtained from monitoring user actions to build up a set of user preferences that reflect the user's needs and wishes as accurately as possible. This problem is more difficult than it would appear, for two reasons:

(1) Their dependence on context. The selections that a user makes or the actions that he/she takes are often dependent on context. For example, when the user is at home, his/her preferences relating to a particular service or device may be different from when he/she is at work or relaxing with friends. Thus the task of building up a preference to cover all possible contexts in general may never be completed.

(2) Changeability of user preferences. Although some preferences may remain constant over lengthy periods, others will not. There are many reasons for this: services may change, new services may become available or the user may simply become aware of existing services that he/she may prefer, a user's circumstances may change, and so on. Whatever the reason, the learning system needs to be able to adapt to such changes.

It is important to involve the user whenever a decision is taken on his/her behalf. This can be done by informing the user whenever the system takes such an action and providing the user with the means to override this if it is not what he/she wants to happen. If the user accepts the action without intervention, this can be taken as reinforcing the preference. If not, the system must inform the learning process. But one is still left with the problem of knowing whether a preference has changed permanently, is subject to a one-off change or a new context situation has arisen.

Thus the problem that is addressed by this paper is how best to acquire a set of user preferences that accurately reflects the user's needs and wishes at any point in time. To try to solve this problem, we are using two different systems – one based on rule-based preferences, the other on neural networks. Both are used in parallel and their outputs are used to determine what action should be taken.

This paper provides some detail on the components of the system and how it deals with this problem.

III. BACKGROUND

Although the basic concepts of ubiquitous/pervasive systems are generally agreed, there are many different scenarios in which they may be used. As a result different researchers have focused on different aspects of the problem or different approaches to solving it, and the details of what should be included in individual systems are still under debate. As a consequence a number of different architectures and prototypes for such systems have emerged, based on different assumptions or different approaches to meeting them.

At one end of the spectrum one type of system in which there has been considerable interest is that concerned with "fixed smart spaces". Much of the original work in this area focused on the "Smart Home". There has been considerable interest in the development of systems that would control the devices and services available in a smart home, especially to provide support for elderly and disabled residents, making it safe for them to live at home. The ideas explored have ranged from simple home automation to more sophisticated domestic ubiquitous computing environments. Examples of systems of this type include the Adaptive House [2], MavHome [3], Synapse [4], Ubisec [5], the Intelligent Home [6], etc.

In addition to work on the smart home, much research has been devoted to developing intelligent systems to control devices for other types of buildings. For example, building automation within commercial buildings can cover lighting, heating, ventilation, security, communication, and other systems. This may be based on a single room or on a whole building. Once again the ideas explored have ranged from simple building automation to more sophisticated intelligent buildings. Obvious examples include work done on the smart office and on smart buildings for large organisations or public buildings. For example, MIT's Project Oxygen [7] uses collections of embedded devices to "create intelligent spaces inside offices, buildings, homes and vehicles".

Besides the work done on fixed smart spaces there has also been considerable interest in developing systems to support the mobile user. The problems here are different from those of fixed smart spaces, and at least as challenging. A number of research projects have explored pervasive system architectures for the mobile user and developed prototypes to demonstrate these. Examples include Mobilife [8], Spice [9], Daidalos [10], etc. The latter project explored two separate architectures for pervasive systems, focusing particularly on mobile users, and developed prototypes for each of these.

In both fixed and mobile systems the main aim has been to relieve the user of some of the burden of detailed interaction and decision making that is needed. To do this, it is essential that the system is aware of the needs and preferences of the user and uses these to take decisions on the user's behalf. This process (of creating, maintaining and applying user preferences in decision making) is sometimes referred to as personalization since it has the effect of tailoring the system's behaviour to the particular needs and wishes of the individual user so that it appears or acts differently for different users or for the same user under different circumstances.

Thus far the prototypes that have been developed for different ubiquitous/pervasive systems have adopted different approaches to personalization. Some of the early developments focused on using context information rather than user preferences, and the resulting systems displayed context aware rather than personalized behaviour. However, the importance of incorporating user preferences into the decision making was recognised and most projects now incorporate both context awareness and some form of personalization.

One important decision that developers are faced with regarding user preferences is the approach used to represent and evaluate them. The two main contenders for this are some form of rule-based representation or an artificial neural network approach. The former has the advantage that the preferences can be viewed by the user and altered manually if the user disagrees with them while the latter has advantages in terms of adaptability.

As pointed out in the previous section, the most challenging problem with personalization lies in the acquisition of the user preferences. The simplest way of doing this is to monitor the user's behaviour and use different forms of machine learning to extract the preference information. However, in doing so one needs to find solutions that address the problems of incompleteness (due to missing context) and changeability (distinguishing real changes to the user preferences from one-off situations affecting the user's choices).

IV. Personal Smart Spaces

Much of the research done on the development of pervasive systems has been focused on fixed smart spaces. Such systems can provide significant levels of support for a user who is within the fixed space although, unfortunately, when the user steps outside of it, the support disappears. This situation has been described as giving rise to "islands of pervasiveness" surrounded by a void in which support for pervasive behaviour is at best limited.

This problem was the main motivation behind the Persist project [11]. The latter was a large European research project which aimed at developing a pervasive system that bridged the gap between fixed smart spaces and mobile systems. The approach that was developed, used the notion of a Personal Smart Space (PSS) to combine fixed smart spaces with those of mobile systems. The result is a new type of system in which the user is constantly covered by their own pervasive PSS. However, while this does ensure continuous coverage for the user, the behaviour of the system at any point in time and the facilities it can provide will depend on whatever other PSSs are nearby.

In order to create a PSS, the devices belonging to a single user together with a set of services that are owned, controlled or administered by the user, are connected together in a network in such a way as to behave like a single system. This set of devices form a dynamic space in which individual devices can join or leave as required. They are connected using peer-to-peer communication and operate together as a unit (although they can operate independently if required). Likewise the set of services associated with the PSS can be shared or withheld according to the current context. One major advantage of this approach is that it can be achieved without the need for any fixed infrastructure (although it is able to make use of such infrastructure where it is present).

Another important aspect of a PSS is the fact that it may be either fixed or mobile. Thus the PSS belonging to a person will generally be a mobile pervasive system that moves around with the user and provides him/her with control over the set of devices and services that form part of the PSS at any point in time. On the other hand a fixed or static PSS is located in a fixed position and provides access to the devices and services that fall under its control. This is equivalent to the fixed smart space generally associated with a smart building of some form (smart home, smart office, etc.).

When the mobile PSS belonging to a user approaches another PSS (either fixed or mobile), the two PSSs communicate with each other, each sending a message to identify itself to the other. Each PSS can then check whether it recognizes the other PSS and can determine the level of trust it has in the other. On the basis of this it will determine whether and how much information about itself it is prepared to share with the other PSS and whether or not to make available some of its services to the PSS. This applies whether the other PSS is mobile or fixed.

In the case of Persist, the architecture used for developing PSSs [12] is shown in Fig. 1. The same architecture applies to both fixed and mobile PSSs.

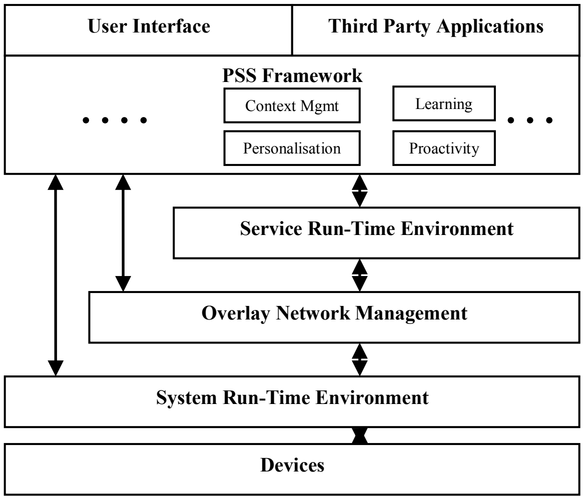

Fig. 1. The high level architecture of a Personal Smart Space in Persist

V. MOBILE AND FIXED SMART SPACES

From the previous section it was noted that each user is associated with his/her own mobile PSS which moves around with the user. At the same time a fixed smart space may be controlled by a fixed PSS, which operates in exactly the same way as the mobile PSS. As a mobile PSS moves around and comes into range of another PSS, it will attempt to establish communication with it in a peer-to-peer fashion. If they are successful the two will be connected in a common network. As other PSSs come within range, they may join the network, and as they move away they will detach from the network. The result is a completely dynamic network.

In the case of a fixed PSS, the same applies. When a mobile PSS approaches it, they establish appropriate communication between themselves via a common network. If other PSSs approach they connect to the same network, and detach from it if they move away.

Based on this there are two important properties of a PSS affecting its behaviour. The first of these is that of personalization, i.e. the ability of the system to keep track of the individual preferences of its owner and to use these to adapt the behaviour of the services it runs. This means that each PSS may behave slightly differently from any other.

The second important property of a PSS is its ability to offer services to other PSSs. This is particularly important for a fixed PSS. Thus the PSS associated with an intelligent building such as a smart office or smart home may offer a variety of different services such as those relating to environmental control. Thus in the case of a smart home or smart office the system may provide a set of services to control room temperature (via heating or air conditioning), ambient lighting (through lights and curtains or blinds) and ventilation. Each of these services may have one or more parameters that are used to convey user preferences to the service. Thus the fixed PSS will communicate with the PSSs of the users within it and offer each of them an appropriate set of services.

By way of illustration, consider a situation in which a smart office has a PSS which controls a number of devices in the office, including heating and lighting. When no one is present in the office (i.e. no other PSS is present), the preferences of the smart office PSS itself are applied. These may be to minimize energy consumption by switching off all lighting, heating and air conditioning if no one is present. When a mobile PSS corresponding to a recognized user enters the office, the office advertises the temperature and lighting services to the mobile PSS, which in turn returns its preferences. These are then communicated to the services via the appropriate parameters, and the services respond by adjusting the temperature and lighting to meet the user's preferences.

If more than one mobile PSS corresponding to a recognized user is present and their preferences in relation to temperature and lighting are the same, the services will respond accordingly. But if more than one PSS is present and the corresponding preferences are not identical, this represents a conflict and an appropriate conflict resolution mechanism is invoked. This may be based on priority (e.g. most senior person present, disabled person with particular needs for temperature control, etc.), some form of averaging procedure or other appropriate mechanism.

On the other hand suppose that the mobile PSS entering the room belongs to a cleaner who enters the room after hours. Once again the PSS of the cleaner and the room PSS will establish communication between them. However, in this case, even though the cleaner's PSS may have a preferred temperature, the room PSS may not provide the cleaner's PSS with access to the services responsible for temperature control and the cleaner's preferences in this regard will be ignored. On the other hand, the room PSS will still offer the cleaner's PSS access to the lighting services.

This approach has several advantages. In the first place the mobile PSS is responsible for maintaining the preferences of its owner rather than having to build these into the fixed PSS. This means that the user's preferences are kept in one PSS under the user's control rather than having them duplicated in different fixed PSSs over which the user has little or no control. This is also of benefit to the learning process which can use the occurrence of similar services in different fixed PSSs to build up a more accurate profile of user preferences more rapidly.

In the second place the fixed PSS controls the processes of making services available to mobile PSSs and can decide which services to offer to any particular PSS (e.g. office worker vs cleaner). This way of separating functionality between fixed and mobile PSSs places the burden of conflict resolution on the fixed PSS which is responsible for the device.

The idea of the PSS has been extended in the SOCIETIES project to that of a CSS.

VI. PERSONALISATION IN SOCIETIES

In the prototype system that we have been developing for SOCIETIES, personalization may be used for a variety of tasks, including:

- Service personalization. Any service may have any number of 'personalizable parameters' which may be set by the action of a preference outcome.

- Virtual identity selection. In order to maintain user privacy, the user may have a number of virtual identities, and user preferences may be used to select the most appropriate virtual identity for a particular service in a particular context.
- Other system personalization. Miscellaneous other system components may make use of preferences to tailor the overall service to the user.

To handle this, the system needs to build up a picture of the user's preferences. To do this, it must be capable of the following:

(a) Monitoring user behaviour. It must recognize relevant user actions and the context in which they are made.

(b) Learning behaviour patterns. From the accumulated data the system needs to be able to extract/learn patterns of behaviour of the user.

(c) Applying behaviour patterns. The system needs to be able to recognize when a situation arises which matches the conditions of a particular behaviour pattern, and apply this pattern to reproduce the appropriate user action on behalf of the user.

In the system that is being developed in SOCIETIES we are using both a rule-based approach and a neural net approach together to improve the overall accuracy and adaptability of the system.

VII. RULE-BASED APPROACH

The rule-based approach was selected for two reasons. First the data obtained from monitoring user actions and the context in which they occur, can easily be stored in a database; then at appropriate points in time machine learning techniques can be applied to this data to identify user preferences in the form of rules (or decision trees). In the second place, preference rules in this form can be viewed and understood by the user. If the user disagrees with any rule, he/she can alter them manually, thereby giving the user ultimate control over the ways in which their environment is adapted.

Since many user preferences are context-dependent, it is natural to use an IF-THEN-ELSE format – in our case, a nested IF-THEN-ELSE format. The condition part of each IF-THEN-ELSE contains conditions based on user context. The result of executing such a rule is referred to as the outcome, and represents an action that the system needs to perform. An example is:

IF service = VoIP AND location = work
THEN ringtone = tune1

For each of the types of personalization outlined in the previous section the preferences have the same format. This not only makes it easier for the user to understand but also for the system to create and manage.

In our system the user always starts off with an initial default preference set. This could simply be a standard default set or one could provide different default subsets for different types of users, i.e. some form of stereotyping. Whatever the case, this initial set merely provides a starting point which is adapted with time as the individual user's preferences become known. In the process existing preferences may be altered or refined while new preferences may be discovered and added.

As mentioned the process of refining existing rule-based preferences and acquiring new ones is achieved through monitoring user actions and inferring preferences through some form of machine learning. The type of action that is referred to here is any act performed by the user that changes the behaviour of a service – whether an internal service of the PSN system or an external third party service. Thus the first step is to identify the particular types of action that are needed for user preferences.

The component responsible for User Monitoring is alerted whenever an action of the type referred to is identified. The information about the action is then stored together with the relevant context information in the History database. The crucial challenge here lies in selecting "relevant" context as storing the complete set of context attributes each time an action is encountered would lead to huge storage requirements and a significant increase in processing requirements while most of the context data would not be relevant.

One approach which helps to reduce the problem is to identify groups of actions that have the same or similar sets of relevant context attributes. However, ultimately, this challenge of distinguishing what context attributes are relevant for what actions, rests with the system developer to resolve.

The algorithm which we use for inferring preferences from the History database is based on C4.5. Gain ratios are used instead of simple Gain to avoid any problems that might arise from attributes with multiple values. The algorithm has also been adapted to include the calculation of confidence levels that are used in subsequent preference merging and conflict resolution.

However, this can lead to several problems as the size of the History database grows. As a result a different strategy has been adopted in which the database is divided into two partitions corresponding to short-term and long-term memory.

The short-term memory store is used to contain the set of tuples (user action + context) that have been captured since the last execution of the learning algorithm. When the next execution of the learning algorithm is triggered it is the data in the short-term memory that are used for this purpose. The preferences obtained from this are then merged with the existing preference set to produce an updated set. The data in the short-term memory data set are then added to the long-term memory data set which contains the complete set of data for the user (or an appropriate subset thereof).

If a conflict arises when merging the new preferences with the existing preferences, the complete data set can be used to resolve such conflicts. For example, suppose that one has a user preference

IF service = VoIP AND location = work
THEN ringtone = tune1

and the next execution of the learning algorithm applied to the short-term memory yields

IF service = VoIP AND location = home
THEN ringtone = tune2

these two rules can be merged to produce

IF service = VoIP
THEN IF location = work THEN ringtone = tune1
ELSE IF location = home THEN ringtone = tune2

However, suppose that the user had switched the device to mute while in a meeting at work, then the next execution of the learning algorithm operating only on the short term memory might produce:

IF service = VoIP and location = work
THEN ringtone = mute

When the system attempts to merge this with the existing preference rule, this clearly gives rise to a conflict. It may be that the user has changed his/her mind or it may be that some other attribute of context has not been taken into account. By running the learning algorithm again on the complete data set (long term memory to which short term memory has now been added), one may end up with

IF service = VoIP AND location = work
THEN IF current_task = meeting
THEN ringtone = mute ELSE ringtone = tune1

This illustrates how the preference rules develop from the two data sets.

This two-level strategy has three advantages:

(1) It enables stronger emphasis to be placed on recency through the short term memory store.

(2) It is a much more efficient process in that the time taken to process short term memory is very much less than that required to process long term memory, and, provided no conflict arises, processing short term memory is all that is required.

(3) The long term memory provides the means to resolve conflicts when they do occur, ensuring a better outcome.

VIII. Neural Net Approach

The neural net approach has a major advantage over rule-based systems in that it operates incrementally, using every relevant event either to reinforce learning or to predict actions. As a result it adapts very quickly to changes. In the case of SOCIETIES we are using a fairly simple neural network which takes as input real world values relating to the user's context and the selected preference outcomes, and learns associations between them. It can be visualized as two layers of nodes with weighted connections between them, as illustrated in Fig. 2.

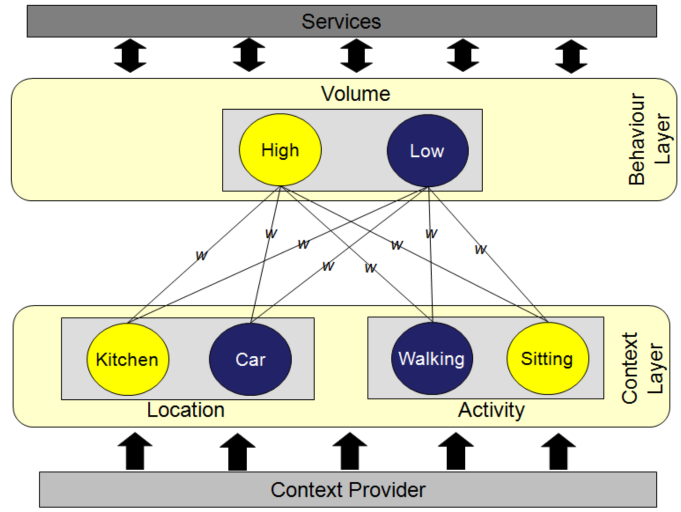

Fig. 2. The topology of the neural network employed in the SOCIETIES Pervasive Social Networking system

This is described more fully in a separate paper [14].

IX. Combined Approach

In the prototype of the SOCIETIES system the two approaches are used in parallel. Whenever a relevant event or change in context occurs, both systems are activated, and used to predict what action the system needs to take to personalize its behaviour at any point in time. If the results produced by the two techniques agree, the system proceeds with the action.

If the results produced by the two techniques do not agree, control is passed to a separate procedure which handles conflict resolution. This uses several bits of information to decide which result to act on, including the Confidence level and Date of last update associated with the Preference Rules. Whatever the final decision, the user is informed so that he/she can intervene and change the decision if it is not what is wanted.

X. Conclusion

The SOCIETIES project aims to combine the ideas of pervasive computing with those of social networking systems to produce a Pervasive Social Networking system (PSN) with the properties of both. In order to create such a system in which full pervasive system behaviour is combined with social networking functionality, the approach we have followed is to build a pervasive system with its own social networking functionality which can connect to and interact with other existing social networking systems. This builds on the notion of a Personal Smart Space (PSS) developed within the Persist project and extends this to incorporate social networking concepts.

The platform that is being developed will be evaluated in a set of three separate user trials during the next six months. These are:

(1) Student trial, in which a number of students will be given the platform to use over an extended period of several months.

(2) Disaster management trial, in which the system will be employed by real disaster management end users in simulated disaster situations.

(3) Enterprise trial, in which the system will be used by industrial users for typical situations in commerce and industry, including conference type applications.

However, in order for the system to be exposed to real users, it must have a reliable system for handling user preferences, including dealing with the problems of incomplete preference rules due to context dependency and of changing preferences.

This paper describes how the combination of rule-based preferences with neural nets is being used to improve accuracy and adaptability in the SOCIETIES system.

ACKNOWLEDGMENT

This work is supported by the European Union under the FP7 programme (SOCIETIES project) which the authors gratefully acknowledge. The authors wish to thank all colleagues in the SOCIETIES and Persist projects, without whom this would not have been possible. However, it should be noted that this paper expresses the authors' personal views, which are not necessarily those of the SOCIETIES consortium. Apart from funding the SOCIETIES project, the European Commission has no responsibility for the content of this paper.

REFERENCES

[1] M. Satyanarayanan, "Pervasive computing: vision and challenges," IEEE PCM, vol. 8(4), pp. 10—17, 2001.

[2] M.C. Mozer, "Lessons from an adaptive house," in D. Cook and R. Das (Eds.), Smart environments: technologies, protocols and applications, pp. 273-294, 2004.

[3] M.G. Youngblood, L.B. Holder and D.J. Cook, "Managing adaptive versatile environments," Proc. 3rd IEEE Int. Conf. on Pervasive Computing and Communications (PerCom '05), pp. 351-360, 2005.

[4] H.K.Y. Si, "A stochastic approach for creating context-aware services on context histories in smart home," Proc. ECHISE2005, Pervasive '05, pp. 37-41, 2005.

[5] J. Groppe and W. Mueller, "Profile management technology for smart customizations in private home applications," Proc. 16th Int. Workshop on Database and Expert Systems Applications (DEXA '05), pp. 226-230, 2005.

[6] V. Lesser, M. Atighetchi, B. Benyo, B. Horling, A. Raja, R. Vincent, T. Wagner, P. Xuan, and S. Zhang, "XQ: the intelligent home testbed," Proc. Anatomy Control Software Workshop (Autonomous Agent Workshop), pp. 291-298, 1999.

[7] L. Rudolph, "Project Oxygen: pervasive, human-centric computing—an initial experience," Proc. 13th Int. Conf. on Advanced Information Systems Engineering, Springer, pp. 1–12, 2001.

[8] M. Strutterer, O. Coutand, O. Droegehorn, and K. David, "Managing and delivering context-dependent user preferences in ubiquitous computing environments," Proc. Int. Symp. on Applications and the Internet Workshops (SAINTW '07), 2007.

[9] C. Cordier, F. Carrez, H. Van Kranenburg, C. Licciardi, J. Van der Meer, A. Spedalieri, J. P. Le Rouzic, and J. Zoric, "Addressing the challenges of beyond 3G service delivery: the SPICE service platform," Proc. Workshop on Applications and Services in Wireless Networks (ASWN '06), 2006.

[10] M.H. Williams, N.K. Taylor, I. Roussaki, P. Robertson, B. Farshchian, and K. Doolin, "Developing a pervasive system for a mobile environment," Proc eChallenges 2006 – Exploiting the Knowledge Economy, IOS Press, pp. 1695 – 1702, 2006.

[11] M. Crotty, N. Taylor, H. Williams, K. Frank, I. Roussaki, and M. Roddy, "A pervasive environment based on personal self-improving smart spaces," Proc. Ambient Intelligence 2008, Springer-Verlag, Heidelberg, 58 - 62, 2009.

[12] E. Papadopoulou, S. Gallacher, N.K. Taylor, and M.H. Williams, "A personal smart space approach to realising ambient ecologies," Personal and Mobile Computing, vol. 8, pp. 485-499, 2012.

[13] S. Gallacher, E. Papadopoulou, N.K. Taylor, F.R. Blackmun, and M.H. Williams, "Intelligent systems that combine pervasive computing and social networking," Proc Ninth International Conference on Ubiquitous Intelligence and Computing (IEEE UIC 2012), IEEE Computer Society, pp. 151-158, 2012.

[14] S. Gallacher, E. Papadopoulou, N.K. Taylor and M.H. Williams, "Learning user preferences for adaptive pervasive environments: an incremental and temporal approach," ACM Transactions on Autonomous and Adaptive Systems (TAAS), vol. 8(1), Article 5, 2013.