6. Dynamic Configuration of Home Services

Tony MCBRYAN and Phil GRAY
School of Computing Science, Sir Alwyn Williams Building, Lilybank Gardens, Glasgow, G12 8QQ, Scotland

Abstract. This chapter addresses the challenge of the configuration of modern interactive systems (ubiquitous, context-sensitive, mobile et al.) where it is difficult or impossible to predict the resources available for evolution, the criteria for judging the success of the evolution, and the degree to which human judgements must be involved in the evaluation process used to determine the configuration. A conceptual model of interactive system configuration over time is presented (known as *interaction evolution*) which relies upon the following steps; (i) identification of opportunities for change in a system, (ii) reflection on the available configuration alternatives, (iii) decision-making and (iv) implementation, and finally iteration of the process. This conceptual model underpins the development of a dynamic evolution environment based on a notion of *configuration evaluation functions* (hereafter referred to as *evaluation functions*) that provides greater flexibility than current solutions and, when supported by appropriate tools, can provide a richer set of evaluation techniques and features that are difficult or impossible to implement in current systems. Specifically this approach has support for changes to the approach, style or mode of use for configuration. These features may result in more effective systems, less effort involved to configure them, and a greater degree of control for the user.

Keywords. Generic, Evolution, Interaction

1. Introduction

This chapter provides guidance to those interested in designing and building home care systems which rely upon a variety of interaction modalities using a multitude of devices such as those described in Chapter 7 of this book. The work in this chapter is specifically aimed at addressing issues that arise when it is not possible to determine which of these devices will be available, when they will be available or how they should be used within the home. This chapter is aimed at those with a computing science or software development background although some of the earlier sections may be of interest to home care and tele-health care practitioners with an interest in the considerations made during the design of home care systems.

Ubiquitous systems typically use large numbers of sensors to detect the state of the environment of use [1] and offer multiple different devices and methods of interacting with users [2]. The multiplicity and volatility of these contexts of use, including the presence or absence of devices and resources, especially when the users or devices are mobile, leads to a demand for systems that are capable of extensive and regular reconfiguration with regard to choice of interactive techniques and components. In addition, as the opportunities for reconfiguration grow, so does the likelihood that users will attempt to appropriate their systems to exploit this flexibility, providing new application functionality in new ways.

This situation has led to the development of software architectures and technologies that enable this dynamic reconfiguration to take place, and also to the development of a variety of techniques for carrying out this configuration. The latter range from conventional preference settings through interactive configuration interfaces to autonomic context-sensitive systems that adjust the form of interaction to the current state of the user and setting. This may be based on sophisticated policies or via matching to previous similar patterns of use. Each of these techniques is useful in certain circumstances and, indeed, combinations of the techniques are also possible.

From both a design and an implementation point of view, it would be desirable to treat all of these techniques in a unified way, as variants of a single coherent model of configuration, so that they can be more easily compared, transformed, combined, refined and swapped. This paper presents such a model, based on the notions of configuration possibilities and evaluation functions over them. We shall argue that this model offers a rich design space for a range of configurations, making it easier to combine techniques and to develop new variants of existing ones.

2. Background

Many techniques for choosing an appropriate interaction technique or device have been developed in the context of ubiquitous systems design. In this section we summarise some of the most popular approaches with some exemplar implementations. This section is intended to discuss the use of the system from the perspective of a typical user and does not compare architectural features of particular approaches.

Kelly and Keenan [3] discuss the application of agile development methods to the development of home care systems. Agile development involves an interative approach to software development through the completion of user stories which represent high level requests from the users. Subsequently as each user story is developed a new version of the software can be delivered to the stakeholders and the existing stories can be revisited after stakeholders have had some experience with the system. However, this obviously requires ongoing collaboration between the developers of the system and the users and does not permit for extensive individualisation of the system as modification is carried out by software developers alone.

Thevenin and Coutaz [4] present the notion of plasticity that identifies equivalence of usability as the primary criterion for assessing interaction adaptation. Their implementation demonstrates automatic and semi-automatic generation of user interfaces exhibiting plasticity.

Manual configuration is frequently used to allow the user complete control over a configuration. Using a manual approach it is necessary for the user to specifically make a modification to the configuration when circumstances change. This configuration can be stored in a configuration file, possibly expressed in an appropriate specification language [5] but often commonly manipulated by an interactive editor such as Jigsaw [6] which uses a ‘jigsaw pieces’ metaphor to enable a user to see the interconnection of devices and to manipulate them to meet changes in demand. Another similar approach is Speakeasy [7] that allows direct connections, as in Jigsaw, but also employs a task based approach where templates are filled out with the appropriate devices by the user.

Context-sensitive systems choose the interaction techniques to use based on data gathered from the user’s environment - their context. Schmidt [8] describes a hierarchical model of context which includes the user model(s), social environment, task model, environmental conditions and physical infrastructure from which adaptations are derived.

Another approach is to define a ‘utility function’ that automatically decides which interaction styles or devices should be used to communicate with the user based on criteria chosen by the system developer at design time. These utility functions may then make use of any contextual data gathered as part of the function. This is the approach taken by Sousa and Garlan [9], where a utility function is used to express the combination of the user’s preferences, the supplier’s preferences and quality of service preferences. The task of making a choice is then an effort to maximise this utility function. This approach is also found in Supple [10] which performs user interface adaptation according to a utility rule based on pre-assigned weights for screen components.

Rule based reasoning can be used to select appropriate interaction techniques automatically based on rules or policies manually set by the user. In the work of Connelly and Khalil [11] this takes the form of policies for devices and interaction spaces being combined to determine the interaction methods that are allowed to be used. This approach is also a clear influence on the current work being undertaken by W3C Ubiquitous Web Applications [12] where content and presentation are selected based on the characteristics of the device(s) currently in use. Another approach used by the Comet architecture (COntext of use Mouldable widgET) [13] is to employ introspective components that publish quality of use guarantees for a set of contexts of use. Adaptations are triggered by policies; the current context of use will be derived and compared against the quality of use guarantees published by available Comets to make a decision on which component should be used. Each component must therefore be able to identify its own quality of use statistics in each of the contexts of use it is possible to appear in. It is also possible to use ‘recommender’ or collaborative filtering techniques to make the decision. A recommender algorithm may use a collection of preference or usage histories and compare them to similar information, either from the same user or from multiple users. This approach is used in the Domino system [14] to determine which components a user has access to, using a history of frequently used components from other users.

A final approach to be considered is employed by the ISATINE framework [15] based on the USIXML mark up language. ISATINE is a multi-agent architecture that decomposes the adaptation of a user interface into steps that can be achieved by the user, the system and by other external stakeholders. The user can take control of the adaptation engine by explicitly selecting which adaptation rule to prefer from an adaptation rule pool in order to express the goal of the adaptation more explicitly. However, this does not provide a mechanism to utilise multiple configuration techniques at run-time. All of these techniques are useful in certain circumstances, but currently no system provides a unified method of offering them all, separately or in combination. Our approach, described below, is intended to provide this unification.

3. Sources of Change

3.1. Stakeholders

Adaptive and configurable systems can involve multiple users and/or multiple stakeholders. Within a home environment there are likely to be partners living in the same space. Friends and family living elsewhere may be involved in care or interested in its status. There may be visiting medical personnel such as community nurses and remotely located medical staff, such as a consultant in a clinic that the patient visits [16]. Within an office or working environment, stakeholders may include friends and family who wish to stay in touch during the course of the day, co-workers, management, customers and clients as well as representatives from other companies or government agencies who may visit during the course of the day.

These people are referred to as stakeholders [17] if they have a direct or indirect interest in how the system works, in how the system is used, or in the data it generates or provides. Many stakeholders may need or want to come into contact with the data or devices of a home care system themselves directly, whether in the clients home or remotely. In this case, these stakeholders have to be considered potential end users of the home care system. Stakeholders would include external agencies responsible for designing, installing, maintaining and prescribing the available equipment. They may also be involved in changes in legislation or policy on how the devices or services can be prescribed and used.

It is likely that with multiple occupants, end users and multiple stakeholders that people's needs, perspectives and accountabilities [18,19] will differ. In addition, these might change over time as the condition of the person and the possible behaviours of the systems change. A system's configuration may be acceptable for some but not for others. For example, the user may wish to have messages and alerts presented by speech, but this might be annoying, disruptive or confusing to a visitor if delivered via loudspeakers while they are present. Similarly, information provided on a TV might be disruptive of TV use by others in the household. It might also allow private and potentially embarrassing health information to be seen by others.

This can result in complex, dynamic and potentially conflicting needs and requirements. Therefore methods are needed for identifying, negotiating, and resolving these changing requirements and interaction needs as the stakeholders interact with and use adaptive systems [20].

3.2. Available devices and service

Adaptive systems should be capable of providing implicit, multi-modal, and non-standard means of interacting to facilitate a more natural user experience. This is likely to include the use of speech and non-speech audio [21], graphical output delivered via mobile devices or digital TV, gesture input and tactile output. Allowing users the choice of various modalities for different interaction tasks in different contexts is important [21]. Knowing which combination of these to use at any one time for any one purpose is not straightforward and is subject to change later on as experience with a modality may change how it is used.

New devices and services may become available purely as the person's context or location changes. Presenting information via the TV, for example, makes more sense in the living room or a shared space than in the bathroom or an empty room. Presenting information via a loudspeaker makes more sense if a person who prefers speech output is present and there is no other audio output to that device at that time. As new devices and services become available, the system must be able to accommodate these new approaches and users must be offered ways to interact with these devices and/or services.

3.3. Changing needs and conditions

A person's needs and conditions, within the environment in which an adaptive system is deployed, will change over time. This is especially true of home care systems. Home care systems are particularly interesting as they pose some particular challenges in respect to changing needs over time. Every patient or user of home care systems has a unique lifestyle, living environment and course of life [22] and as such the systems must be able to adapt and change to the users needs over time.

It is common in an ageing population that the people being cared for will have a cluster of medical conditions to manage [23] such as diabetes, strokes, asthma, epilepsy or orthopaedic conditions – some of which might interact with each other. This means that a home care system must be capable of dealing with decisions on which rules to follow if health indicators from different conditions or symptoms conflict with each other. There is, of course, the added problem that conditions may be multiple not only with one person but also with multiple persons living within the home. Users of home care technologies can be of any age and ability, but a large number of users are either elderly or have physical, sensory or cognitive impairments, or some combination of these factors. This results in a user group that should be offered appropriate choices of both traditional and novel methods of interacting with the technology to increase accessibility for those with impairments. Offering choices of modalities and interactions is desirable, and yet not necessarily straightforward to achieve. It is necessary, therefore, that home care systems should be able to support preferences and capabilities that vary among users and also as care needs change.

4. Evolution as a Process

Central to the theme of evolution is the idea that design and development occur concurrently, a notion that has been explored by Fischer et al. [24]. In this work Fischer presented a set of complementary systems for designing a kitchen and discovered that the process of designing the kitchen would happen concurrently with the process of specifying the kitchen; users would frequently revisit and improve or retune their designs as they developed them. It is this style of activity that the process of evolution presented in this section is designed to support.

Given the multiple aspects of change presented in Section 3, adaptive systems should be able to adapt to dynamically changing requirements of the client themselves, other relevant stakeholders and the situation of use. Allowing different users the choice of interaction methods for different tasks in different contexts is important to ensure both usability and acceptability.

Previous work has focused on dealing with short-term changes within a home environment such as context aware systems [25,26] that react to situational changes. There is a gap in the literature of methods for supporting longer term configuration. This chapter reinforces the notion of interaction evolution in a ubiquitous system, particularly over the long term. The concept of evolution used here is influenced by Dourish [27], MacLean [28] and Fickas [29]. Each of these authors identifies the ability to evolve, tailor and design a system by the user as a necessary feature for acceptance of ubiquitous or adaptive systems.

Interaction evolution is broadly defined here as *multiple related instances of interaction configuration (customisation or personalisation) over time that have the goal of changing some aspect of the system's interaction behaviour*. For example, an elderly user might develop a visual impairment (e.g. cataracts) that requires a reduction in dependency on conventional visual displays.

Interaction configurations range from automatically-generated rapid changes based on context to a process of modification driven by regular human reassessments of the system and its effectiveness.

The process of evolution is modelled as one or more potentially linked configurations, each of which consists of the following stages, which occur iteratively:

- identification of opportunities for changing the system - an event occurs which indicates a change in the system or the environment that may allow (or require) a change in configuration to occur
- reflection on alternative choices for change - the options for configuration can be analysed in order to determine which configuration option(s) should be used
- decision-making - decisions on whether the system should or should not be reconfigured (based on the previous reflection on the available options) and, if so, how it should be reconfigured;
- implementation - the chosen configuration is implemented.

Figure 1 shows this process as a spiral. The first configuration (1) shown by a solid line is a configuration that has gone through one and a half iterations. The second (2), indicated with a dotted line, shows another configuration that has only just been identified and whose alternatives are under investigation. As shown in the figure it is possible to have multiple configuration processes under way at the same time at different stages of evolution. Each of these stages is now considered in turn.

At first glance this is similar to the idea of autonomic computing [30], an idea developed by IBM where elements manage themselves and integrate themselves into a system based on high-level orders or policies written by an administrator. However, autonomic computing is based upon the idea of a large number of self-regulating, self-managing components which operate independently within a system but which negotiate in order to cooperate with each other. Each autonomic component monitors itself in the form of a closed control loop similar to the one described in this section. It continually checks to see if there any other autonomic components that it can use which are superior to the ones currently in use. However the configuration loop used in autonomic configuration can affect the configuration of only a single autonomic component. Furthermore, the ability of the component to configure itself is limited by the goals that can be represented internally within the components.

These autonomic systems individually configure themselves while following policies that represent the aims or objectives of the administrator. Interaction between autonomic components is based on the idea of cooperative negotiation [31]. Autonomic

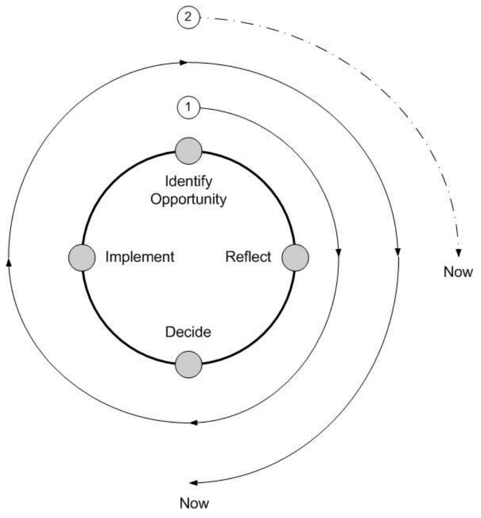

Figure 1. *Process of Interaction Evolution*

components negotiate access to other components directly with these components. Simple forms of negotiation might be ‘first-come, first-served’ where components providing a service will attempt to satisfy all requests until they run into resource limitations. In some cases it might be that only a single request can be satisfied. More complex forms of negotiation include bilateral negotiations over multiple attributes (price, service level, priority) and may involve counteroffers.

However, mechanisms for negotiating, enforcing and reasoning about agreements within autonomic computing are lacking, as are methods for translating them into configurations [30].

Given the home care focus of the MATCH project, each step of this chapters approach is illustrated by the use of a running example taken from the home care domain. The example has been validated by various stakeholders during focus groups and interviews with older users and health and social care professionals [32].

In this example, Fred and Shirley are an older couple with chronic conditions that could be ameliorated by appropriate use of ubiquitous home care technology. In particular, Shirley has worsening arthritis and is no longer able to move around the house easily. She relies on Fred for tasks such as controlling the heating system, closing the curtains and for most household chores. Fred recently had a stroke. He is still physically fit but has become more and more forgetful since the stroke and requires regular reminders for when to take his medication. He is also hard of hearing. They have a daughter Fiona who visits once a week and brings the shopping. A social care worker comes once a week and has offered them additional help with their shopping and household chores, however Shirley and Fred are happy doing things for themselves for now.

4.1. Identify opportunity for change

For an adaptive system to evolve it is necessary to be able to identify opportunities for changing the devices and techniques the system uses to interact with the user. These opportunities are of many types, ranging from rapidly changing circumstances (e.g. ambient noise level) that need a rapid, probably automated change, to slowly emergent conditions that require rigorous (human) analysis and gradual resolution (e.g. deterioration of sight).

Identification of the opportunities for change within a system can include identifying the devices that are available, which are currently in use, which have been added & removed recently in the home care system, as well as the available interaction methods or modality choices.

A candidate for interaction configuration is defined as as a combination of devices, interaction techniques, modalities and supporting components required to instantiate a new configuration in order to satisfy the needs of an application task providing some functionality to the user (reminder notifications, streaming music, controlling devices, etc).

Identification of opportunities for change can take many forms. They can be identified directly by Fred or Shirley, or another involved person, who can then take immediate action to rectify the problem. Alternatively they may gradually become aware of a deficiency in the current configuration before taking steps to address it.

In addition to identification of opportunities for change conducted by Fred and Shirley, it is possible that as new devices, techniques or modalities are added to the system with alternative, potentially better, devices can be substituted into current configurations in an attempt to improve them. This could be detected automatically by the system as it is running - perhaps using aggregate usage data of how other systems have configured this new component in the past, It may be detected that it commonly replaces or is used in conjunction with existing components.

4.2. Reflect / judge alternatives

Once an opportunity for change has been identified, it is necessary to characterise the potential options for taking advantage of it. As with the opportunities themselves, the identification, characterisation and analysis of the options may be straightforward and automatable (e.g. presenting information to the user via the output devices currently nearest to them). However it may be complex, difficult to describe and evaluate (e.g. determining the alternatives for delivering a medical alert to a patient with progressive ocular deterioration) perhaps needing the involvement of experts as well as decision-support tools. Since many systems (including home care systems) are inherently multi-user it may be necessary to support collaboration between various stakeholders and assist in the description.

Once the options for change have been discovered, it is necessary to reason about the available options and to determine their suitability. Some exemplar approaches of reasoning are discussed here with reference to the example scenario presented in this chapter, but there are a number of alternative approaches to reflection and judgement of alternatives within the context of the model.

In an adaptive system, such as in a home care environment, it is likely that users will have preferences for which devices or styles of interaction to use. However in a multi-user environment it is likely that Fred and Shirley will not have the same preferences all the time or in the same circumstances. Thus it is necessary to be able to incorporate multiple criteria into a decision making process on how to accomplish a task.

Fred and Shirley have different capabilities for interaction - Fred has difficulty hearing while Shirley has limited mobility. In this case, speech dialogue interactions may make sense for Shirley as it eliminates problems with physically interacting with a home care system, but this may be an inappropriate choice for Fred. These conditions are likely to change over time and will need to be revisited periodically or when events force a change. This must be supported as an additional interaction within the system.

When a visitor is present, such as daughter Fiona or the social care worker, this contextual change will affect the method of delivering information to the couple. Reminders about medication or household chores may need to be suppressed while other people are present in the home. This problem is exacerbated when the information to be presented to the occupants is of a confidential or embarrassing nature. This requires that contextual information be included in the decision making process.

To support these, and other, types of decision that would need to be made, it is necessary to provide support for several different configuration techniques which allow these decisions. It must be possible for users to be able to manually configure interaction: they are the ultimate arbitrator over a configuration and can have the maximum level of control at the expense of dynamic adaptability. It must be possible to include several analytical reasoning components which operate over the set of possible configurations. Examples of these might be location, preferences or contextual results such as ambient environmental factors which can be directly measured, analysed and decided upon. It must be possible to include techniques which interact with the user on an ongoing basis to maintain relevance as opposed to a 'fire and forget' configuration which would become less appropriate as conditions changed.

It may be possible to assess alternatives based on a record of previous usage (e.g. identifying alternatives that have proved successful or otherwise in similar circumstances). This may be based on logging of user-system interactions or a record of special events of interest (indicators of satisfaction or dissatisfaction) about the current configuration. In addition to using this information to evaluate alternatives it may also form the basis of further evaluation such as trying out new configurations on an experimental basis. Collaborative techniques, such as negotiated choices between interested parties or collaborative filtering [33], are clearly important in a multi-user home. It is necessary to support this ability to allow for conflicting sets of values to be combined in order to decide on the best configuration to use.

Techniques discussed in this section range from fully automatic with no user interaction to those which involve ongoing interaction with the user as the primary concern. Not all of these techniques are appropriate in every situation. The ability to allow for a range of manual and automatic reasoning techniques is a requirement for an effective decision on the correct configuration when making both short and long term changes.

4.3. Make decision

After reflection has taken place it is necessary to make a decision about whether a reconfiguration will take place and, if so, what form it will take.

Both the decision itself and the resulting implementation of the configuration may be deferred until a later time. That is, the opportunity for configuration may be identified and recorded, but the actual configuration does not take place until later. This may be required in situations where the user is currently busy and a change in modality or interaction style would be a distraction from the task at hand. Alternatively, as in the visual deterioration case, the opportunity may be known (e.g. the rate of deterioration may be predictable) resulting in a plan for future reflection and decision making.

Decision-making, like reflection and analysis, may involve multiple agents and hence multiple criteria. Multiple stakeholders might be present in a home care system (such as Fred, Shirley and any visitors to their home). Their independent criteria must be combined to make a choice of interaction, even if they are conflicting or contradictory, so that a solution can be determined.

There are many different benefits and drawbacks resulting from the choice of approach used to combine criteria. Common issues arising from approaches to combining criteria from different sources are preventing dictatorship of one factor, maintaining pareto-efficiency[1] and independence of irrelevant alternatives. However, ubiquitous systems are not necessarily limited by the same constraints that other voting or combination systems are. For example in some situations it may be the case that one criterion actually does matter more than others.

To cope with the issues presented by different types of system it should be possible for multiple such systems to be in use at the same time. This may use combined evaluations incorporating multiple approaches as well as separate evaluations for different interaction tasks using different approaches.

4.4. Implement

Once a decision to make a change has been made it is necessary to transform an abstract decision into an actual change in the configuration of the system. The precise method of doing this will vary among systems. The required steps would be to identify the components that have been newly selected to identify components that should be discontinued, and to arrange the transition that is necessary to switch from one to the other.

Implementation is the ultimate proving ground for a configuration choice. Although it is possible to guess or reason about the suitability of a configuration, it is only by implementing it that it can be discovered if it is truly appropriate for the circumstances.

4.5. Iterate

The entire process of handling change is iterative and ongoing to support evolution of interaction. People do not necessarily know in advance which interaction techniques and devices will and will not work in different circumstances and may need to experiment before deciding. This implies that each iteration should include an evaluation phase as part of identification of opportunities for change. This will determine if the new configuration meets the needs of the users better than it did previously. The users would typically have to be involved in this step to make this judgement.

Over time, new criteria will emerge that will need to be reasoned about in order to choose the best candidate for configuration. Examples of these might include new people or devices moving into the home or a change in the criteria that should be applied. To do this it must be possible to add new techniques or to change the techniques in use

[1]If every criteria prefers option A over option B then option A should always rank higher than option B

within the home. This requires allowing additions or removals from the active models at runtime.

The concept of ongoing re-evaluation, discussed previously, requires a process of evolution to improve the situation over time and through changing circumstances. To accommodate this it must be possible to decide when it is appropriate to perform these evolutionary steps. It may be desirable to change the active configuration as soon as a new device or context change occurs. However in some circumstances it may be desirable to limit the number of changes that take place, or to cause them to occur at a fixed time or after a certain other event has taken place.

5. A Model of Interaction Evolution

The previous section discussed the process of interaction evolution for configuring interactive systems. This followed a series of steps starting from identification of opportunity for change and working through reflection, decision making and implementation of the configuration options in an iterative process.

This section introduces a model-based approach to configuration of interactive systems based on the concept of 'evaluation functions'. This satisfies the requirements of the process while providing a number of compelling features. The approach represents each of the techniques that can be used for configuration within a unified model. It allows designers to provide many configuration techniques in parallel or in combination that are potentially modifiable at run-time and capable of being driven by user interaction.

It is beyond the scope of this chapter to provide technical details on the implementation of this approach. Interested readers are advised to consult McBryan [34] for a comprehensive discussion regarding the implementation of the MATCH framework which is based upon the ideas presented here.

5.1. A Unified Model of Configuration

The model presented here is responsible for identifying opportunities for change as well as reflecting on the alternatives available to make a change.

The concept of a 'configuration possibility' (hereafter, 'possibility') is introduced which is an encapsulated solution (consisting of interaction components, techniques and devices) that can offer interaction between a system task and a user. In terms of the model presented here, the possibility may be composed of functional elements which are abstract concepts, software or hardware. In a software implementation these must be realisable as software or must be able to be interacted with using some software component. Accordingly a possibility includes any software elements needed to perform data transformations related to the interaction, as well as references to the elements that will be responsible for rendering the interaction via physical devices.

Possibility. *A possibility p is defined as an ordered set of size n consisting of components c from the set C (available interaction elements, techniques and devices) that can be used with an application task t from the set of all application tasks T. Formally,* $p = (t, c_1, c_2, c_3 \ldots c_n | c \in C, t \in T)$

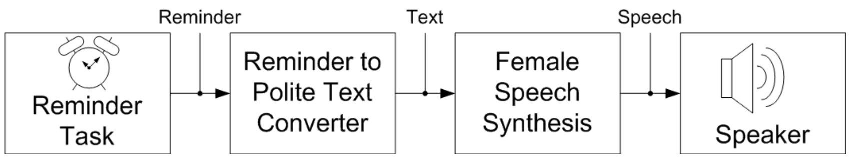

Figure 2. *A typical configuration possibility*

Consider a medication reminder for Fred; One of the possibilities, shown in Figure 2, might be to deliver the reminder via a speech synthesis system. The possibility would include the software representing the physical device (the loudspeaker), the speech synthesis system (responsible for converting text to speech) and the software that converts a medication reminder into the appropriate textual alert.

A key concept in this work is the notion that it is possible to model an interactive system as a directed graph of available components from which the available possibilities can be derived. This graph can be constructed using a service discovery system that models relationships between available components. A directed graph is typically defined on a given set of vertices V (in this case members of the set of components) and edges E.

Possibility Graph. *A possibility graph G is defined as a pair (V,E) where V is a set of vertices ($V \subseteq C$) and E is a set of edges between the vertices $E \subseteq \{(u,v)\ |u,\ v\ \in V\}$*

By identifying interaction components, it is possible to traverse the graph with the goal of constructing a complete set of possibilities that can be used with the application task. This is accomplished by exploring the graph using a recursive graph traversal algorithm to walk each route through the graph.

Traversal. *A traversal is defined as a function (traverse) which operates over a graph (G) parameterised with an application task (t). It returns a set of possibilities (P) such that $P = \{p_1, p_2, p_3 \ldots p_n\}$ where $P\ \leftarrow\ traverse_{(t)}(G)$*

Figure 3 shows a typical, albeit simple, graph that may be constructed from the data in a service discovery system (such as the Ontology Service Discovery described in Chapter . In this graph different possibilities can be deduced (such as the loudspeaker using polite text and a female voice). Also shown is output that requires the choice of two of the intermediate components as well as a GUI that does not require intermediate components and allows the reminder message to be displayed directly. By starting from the reminder task as the root node, a graph traversal can be performed to determine each possibility in the graph.

Once the graph has been built and traversed to create a set of possibilities, it is possible to analyse the appropriateness of each possibility. To do this each possibility is evaluated by using one, or many, evaluation functions.

The purpose of an evaluation function is to rank, filter or otherwise analyse these possibilities to reduce them to a set of selected possibilities which represent a configuration decision that has been made. Evaluation functions can have a many-to-many rela-

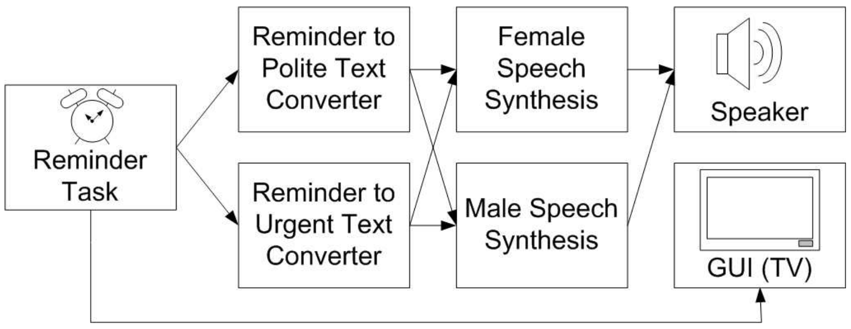

Figure 3. *A typical graph*

tionship with task assignments. There may be many evaluation functions used to review the possibilities for the medication reminder task, and a single evaluation function may be used simultaneously for many tasks. Approval and ranking evaluation functions are used; which allow/disallow possibilities and assign scores to possibilities respectively. Note specifically that the evaluation functions do not necessarily have the same return semantics.

Approval Evaluation Function. *An approval evaluation function (φ) is a restriction (σ) over the set P producing the set P' (a selection of appropriate possibilities from the set of available possibilities). Formally,* $P' \leftarrow \sigma_{\varphi}(P)$

Ranking Evaluation Function. *A ranking evaluation function over the result (P') applies a metric function (M) to score each possibility in P. Therefore,* $P' \leftarrow \{(p_1, m_1), (p_2, m_2)...(p_n, m_n) | p \in P, m \leftarrow M(p)\}$

Figure 4 shows one possible result from the application of two evaluation functions (a ranking and an approval function) to some of the possibilities that have been generated in the previous step. The Usage History Ranking is an example of an evaluation function which uses the recommender approach to rank possibilities. The Doctor's approval function allows or disallows possibilities; here the Male Speech synthesis is disallowed as it sounds too similar to Fred and can confuse Shirley.

To allow multiple evaluation functions to be used with a single task it is possible to use evaluation functions to combine results via function compositions (in effect a 'meta-evaluation function'). This allows the results of multiple approaches (implemented as evaluation functions) to be combined into a single function that can be mapped onto the task.

Meta-Evaluation Function. *A meta-evaluation function is any function (F) which produces a new set of possibilities (P') by combining the results of two or more existing sets of possibilities ($P_1, P_2 \ldots P_n$). Specifically,* $P' \leftarrow F(P_1, P_2 \ldots P_n)$

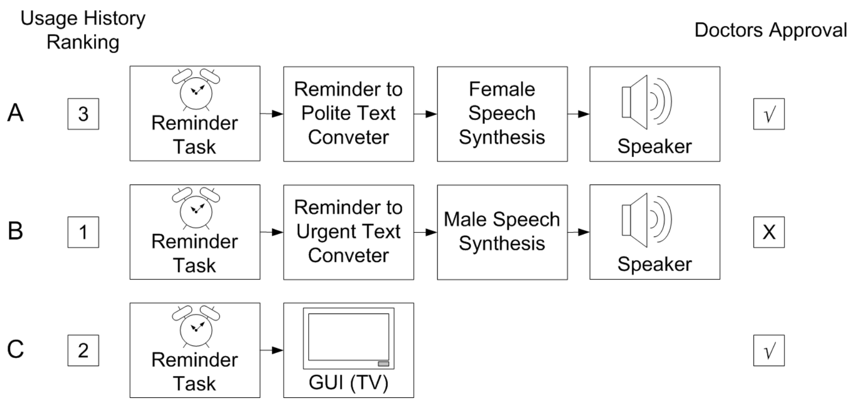

Figure 4. *Example results from the application of a ranking evaluation function and an approval evaluation function*

This approach would allow, for example, the selection of an interaction technique for the notification task to be based on a combination of context-sensitive, manual and/or automatic reasoning. A typical example of this might be that the users' preferences are weighted against the results of a collaborative filtering system receiving input from multiple users, based on the success of similar tasks.

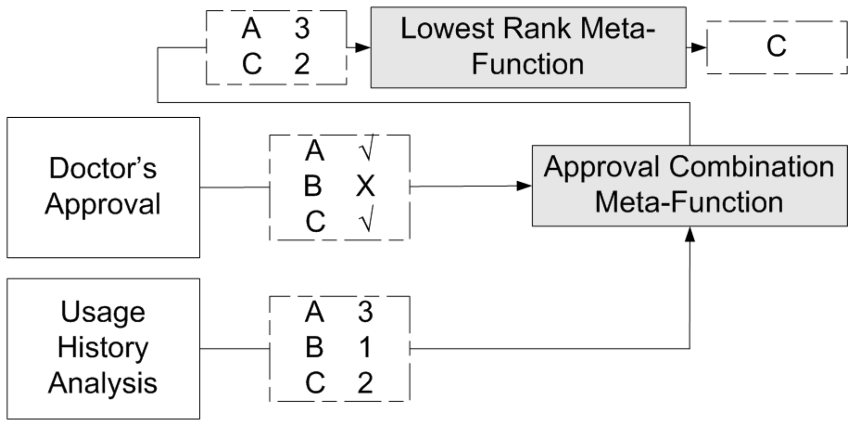

Figure 5. *Example results from the combination of two evaluation functions*

Figure 5 shows one possible method by which the previous two evaluation functions (one ranking and one approval) might be combined to determine which possibility to use from the three available possibilities shown in Figure 4. As the Usage History Analysis evaluation function is implemented as a ranking function it scores each of the possibilities and the results are combined with the results of the doctor's approval evaluation function by a meta-function which removes any of the ranked possibilities which were not also approved. A final meta-function selects the possibility with the lowest numerical rank (corresponding to its position in the rankings). Possibility ‘C’ is the possibility with the lowest numerical rank that was approved and was therefore selected.

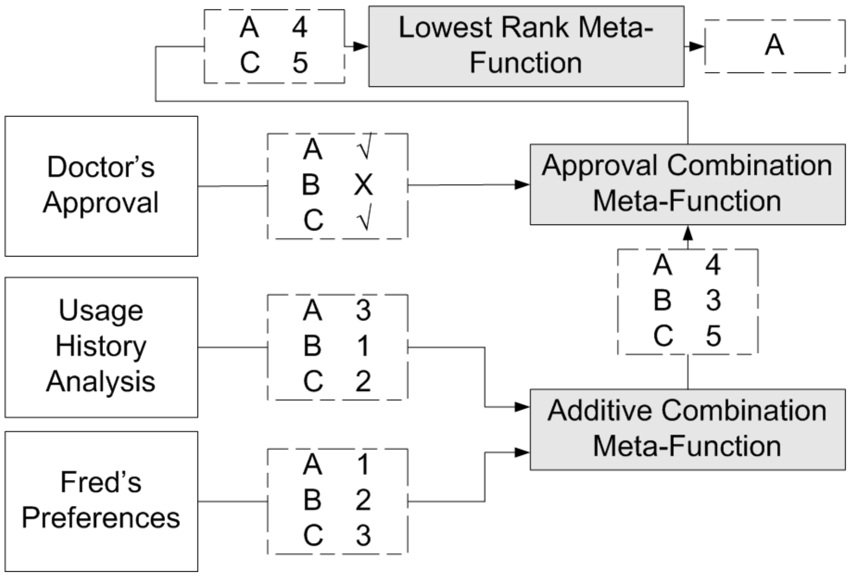

Figure 6. *Example results from the combination of three evaluation functions.*

Functions can be replaced or changed at will to provide different results. Figure 6 shows how introducing a third evaluation function into the tree might affect the results. The individually ranked results of both ranking functions (Usage History and Fred’s Preferences) use a meta-function which combines the separate numerical rankings, e.g. via weighted combination, to create a new ranking. This is then further combined with the doctor’s approval evaluation function and evaluation proceeds as previously. In this case the selected possibility is now ‘A’ as it has the lowest numerical rank that has also been approved.

One feature of this is that the system has native support for combining multiple criteria within a single evaluation process. Each criterion in the task can have its own evaluation function(s) modelled after that criteria’s individual requirements, the results of which can then be combined within the same framework. This allows the natural specification of how criteria can be combined by changing the meta-evaluation function being used to combine the results.

The eventual result of an evaluation function (or tree of evaluation functions) should be the set of possibilities to use for interaction, as shown in Figures 5 and 6. In this case, a single technique has been selected, although functions might enable multiple concurrent techniques to be used.

Evaluation functions are a flexible method of reasoning about the available possibilities and can be applied at different levels of granularity. Some evaluation functions may consider an entire possibility while others may operate only over selected portions of a possibility. For example, an evaluation function may consider only the choice of physical output device in its reasoning. Evaluation functions may utilise external sources of data such as context or usage history and can be parameterised such that a single evaluation function may be reused in multiple situations (such as gathering of user preferences from multiple stakeholders) or even called recursively.

It should be clear that, although the example used here is based on a single output of a single application task, this model is generalisable to inputs as well as outputs, multiple instances of the same application task and to multiple disparate tasks.

6. Conclusion

6.1. Process

This chapter has presented a process which encompasses the primary concepts within interaction evolution as a four-stage iterative process. This is a continual approach and is applied on an ongoing basis. This allows for an iterative decision making process to support evolution of interaction. This design for system support builds upon the ideas for evolution as multiple related instances of personalization or customization. It allows for multiple instances of reflection and judging alternatives. It is capable of individually reasoning over the candidates for evolution which are then combined to allow decision making to take place.

The key features to ask how a system can be controlled is to ask:

- What is the system currently doing?
- What can it do?
- How can it be changed?

The process that has been presented has the minimum necessary steps to address each of these questions.

Identification of opportunities for change - This is required to address the question *How can it be changed?* The enumeration of choices of how a system can change depends upon being able to identify the components that can be changed within a configuration at the appropriate time.

Reflect / judge alternatives - Without the ability to reflect upon the available opportunities for change it is not possible to ask *What can it do?* This implies an ability to differentiate the available options and to reason about the available opportunities in a way which allows them to be selected.

Make decision - The need to decide on what the system should do is derived from the questions *What can it do?* and *How can it be changed?* These questions both imply that one or more of the things it can do will be selected before asking how the configuration can be changed to this selection.

Implement - Once such a decision is made it is necessary to actually implement it. Without being able to do so, determining *What the system is currently doing?* is not possible as the current configuration cannot exist without it being implemented at some previous point in time.

Iterate - Iteration is important as each of these questions can be asked repeatedly. Once a new configuration has been made it is still possible to again ask these questions of the new system configuration. This requires an acknowledgement of the iteration within the process.

As should be evident, the process is best viewed as a collaborative activity involving multiple human stakeholders interacting with the system itself (the target of change) and computer-based support tools in order to make decisions. For this reason, an approach that attempts to link these aspects via a common model is necessary.

6.2. Model

A model has been presented which is composed of a number of key components. These are: (i) possibilities which encapsulate a particular implementable configuration, (ii) a graph representation of elements within possibilities, (iii) an ability to traverse this graph and obtain sets or lists of possibilities as a result, (iv) evaluation functions which can reason over these sets of possibilities, and (v) an evaluation function tree structure which allows for combination of disparate approaches. The execution of this model results in a determination of which possibilities should be implemented.

The model presented in this chapter supports each stage of the process as follows:

Identify opportunities for change: These opportunities can occur in a number of ways within the model. The most obvious is reconfiguration of the graph (or the source of the graph) by addition, removal or modification of any of its entities. However, this is only the coarsest form of identification available. Recall from the examples in this chapter that evaluation functions can respond to external events or incorporate external conditions into an evaluation.

Reflect / judge alternatives: The model proposed in this chapter is specifically designed to allow for extensive reflection over alternatives and to allow this reflection to be orchestrated by computational methods by human interaction or by some combination of these. Specifically, evaluation functions implement a unit of reflection or analysis which can be performed.

Make decision: Likewise, the decision making process is explicitly modelled by the ability to combine evaluation functions (units of reflection) to arrive at a decision about which configurations should be implemented. As this decision process is a combination of evaluation functions it can incorporate any number of varied approaches, combinatorial or human centred, to arrive at the final decision.

Implement: Implementation of the selected configuration is accomplished by the explicit representation of a unit of configuration for a task in the form of possibilities. Possibilities contain all the information necessary for the system to implement a particular configuration.

Iterate / repeat: Finally, the process allows these steps to take place in an iterative and ongoing process. The model presented here allows for a large number of temporal behaviours. Evaluation functions can be queried in an on-demand or timed approach, or they can proactively signal that the criteria they are monitoring have changed and that a global reevaluation is necessary.

6.3. Deployment

The work in this chapter was implemented as part of the MATCH framework. The MATCH framework is OSGi [35] based and allows for 'bundles' to be dynamically loaded and unloaded at runtime. The MATCH framework then additionally adds support for dynamic selection of interaction methods and devices using the techniques discussed in this chapter. Two live deployments of this framework have been made as exploratory technology probes into the mental models used by home care users to select interaction techniques. These deployments found behaviour consistent with the process described in Section 4.

During these deployments a number of observations were made regarding users configuration behaviour. Specifically, (i) users place a high value on their perceived level of control and ability to perceive and directly inspect the current configuration as well as the rules which have been set, (ii) there is a wide variety of preferences between users in terms of mental models for the most important foci for configuration (inputs vs. outputs, people vs. devices) and (iii) configuration occurs as a combined result of evolutionary changes to test new configurations and immediate changes inspired as a result of the realisation of an incorrect configuration and knowledge of how to fix it.

Readers interested in the methodology and results of these deployments are directed to McBryan [34].

6.4. Overview

This chapter has argued that, due to the dynamic nature of home care, novel methods are required for the development of home care systems. It has detailed the features that characterise home care and illustrates the complexity of the home care domain. It suggests several features that should be available in requirements engineering for home care technology and describes methods for system support for interaction evolution in home care systems.

This chapter has presented a model-based approach to supporting configuration. The approach allows for the combination of multiple techniques ranging from fully automatic to fully interactive approaches for configuration, including various intermediate combinations.

The approach described here expressed composition and function without using a specific specification or description language. Instead supports the combination of multiple disparate languages (for example; Java, ACCENT [36,37], MATLAB) within a single configuration if so desired. The approach is intended to be realised as a tool supported configuration system where evaluation functions can be combined and specified by the stakeholders. However, it may prove useful to express configurations in the model via a custom language.

The examples, described above, involve only the selection and configuration of output components. However it is equally applicable to the selection, combination and configuration of components involving both input and output.

References

[1] A. K. Dey and J. Mankoff. Designing mediation for context-aware applications. In *ACM Transactions on Computer-Human Interaction (TOCHI)*, volume 12, pages 53–80. ACM, 2005.

[2] S. Oviatt. Ten myths of multimodal interaction. *Communications of the ACM*, 42(11):74–81, 1999.

[3] S. Kelly and F. Keenan. Investigating the Suitability of Agile Methods for Requirements Development of Home Care Systems. *J. Software Engineering & Applications*, 3:890–893, 2010.

[4] D. Thevenin and J. Coutaz. Plasticity of User Interfaces: Framework and Research Agenda. In *Proceedings of Interact*, volume 99, pages 110–117, Edinburgh, UK, 1999. IOS Press.

[5] J. Magee, N. Dulay, S. Eisenbach, and J. Kramer. Specifying Distributed Software Architectures. In *Proceedings of the 5th European Software Engineering Conference*, pages 137–153, Barcelona, Spain, 1995. Springer.

[6] J. Humble, A. Crabtree, T. Hemmings, K.-P. Åkesson, B. Koleva, T. Rodden, and P. Hansson. Playing with the Bits – User-configuration of Ubiquitous Domestic Environments. In *Proceedings of the Fifth Annual Conference on Ubiquitous Computing (UbiComp)*, pages 12–15, Seattle, Washington, USA, 2003. Springer.

[7] W. K. Edwards, M. W. Newman, J. Sedivy, T. Smith, and S. Izadi. Challenge: Recombinant Computing and the Speakeasy Approach. In *The 8th Annual International Conference on Mobile Computing (MOBICOM)*, pages 279–286, Atlanta, Georgia, USA, 2002. ACM.

[8] A. Schmidt, M. Beigl, and H.-W. Gellersen. There is more to context than location. *Computers & Graphics*, 23(6):893–901, 1999.

[9] J. P. Sousa and D. Garlan. Improving User-Awareness by Factoring it Out of Applications. In *System Support for Ubiquitous Computing Workshop (UbiSys) at the Fifth Annual Conference on Ubiquitous Computing (UbiComp 2003)*, Seattle, Washington, 2003.

[10] K. Gajos, D. Christianson, R. Hoffmann, T. Shaked, K. Henning, J. J. Long, and D. S. Weld. Fast and robust interface generation for ubiquitous applications. In *Proceedings of the Conference on Ubiquitous Computing (UbiComp)*, pages 37–55, Tokyo, Japan, 2005. Springer.

[11] K. Connelly and A. Khalil. Towards Automatic Device Configuration in Smart Environments. In *System Support for Ubiquitous Computing Workshop (UbiSys) at the Fifth Annual Conference on Ubiquitous Computing (UbiComp 2003)*, Seattle, Washington, 2003.

[12] World Wide Web Consortium (W3C) Ubiquitous Web Applications. Content Selection for Device Independence (DISelect) 1.0. http://www.w3.org/TR/cselection/, 2007.

[13] G. Calvary, J. Coutaz, O. Dâassi, L. Balme, and A. Demeure. Towards a new generation of widgets for supporting software plasticity: the "Comet". In *Preproceedings of Engineering for Human-Computer Interaction / Design, Specification and Verification of Interactive Systems (EHCI/DSV-IS)*, volume 4, pages 41–60, Hamburg, Germany, 2004. Springer.

[14] M. Bell, M. Hall, M. Chalmers, P. Gray, and B. Brown. Domino: Exploring Mobile Collaborative Software Adaptation. In *Pervasive'06. Proceedings of the 4th international conference on Pervasive Computing*, pages 153–168, Dublin, Ireland, 2006. Springer.

[15] V. L. Jaquero, J. Vanderdonckt, F. Montero, and P. Gonzalez. Towards an Extended Model of User Interface Adaptation: The ISATINE framework. In *Engineering Interactive Systems (EIS)*, pages 374–392, Salamanca, Spain, 2007. Springer.

[16] M. R. McGee-Lennon and P. Gray. Including Stakeholders in the Design of Homecare Systems: Identification and Categorization of Complex User Requirements. In *Include Conference*, London, 2007. Royal College of Art.

[17] H. Sharp, A. Finkelstein, and G. Galal. Stakeholder identification in the requirements engineering process. In *Proceedings of 10th International Workshop on Database & Expert Systems Applications (DEXA)*, pages 387–391, Florence, Italy, 1999. IEEE Computer Society Press.

[18] S. Garde and P. Knaup. Requirements engineering in health care: the example of chemotherapy planning in paediatric oncology. *Requirements Engineering*, 11:265–278, 2006.

[19] X. Liu, C. S. Veera, Y. Sun, K. Noquchi, and Y. Kyoya. Priority assessment of software requirements from multiple perspectives. In *28th Annual International Computer Software and Applications Conference*, pages 410–415, Hong Kong, 2004. IEEE Computer Society Press.

[20] M. R. McGee-Lennon and J. S. Clark. Multi-Stakeholder Requirements in Home Care Technology Design. In *Workshop on Distributed Participatory Design. Computer Human Interaction (CHI) conference on Human factors in computing systems*, Florence, Italy, 2008. ACM SIGCHI.

[21] M. R. McGee-Lennon, M. Wolters, and T. McBryan. Audio Reminders in the Home Environment. In *Proceedings of the International Conference on Auditory Display (ICAD)*, pages 437–444, Montreal, Canada, 2007. Schulich School of Music, McGill University.

[22] M. Z. Eslami and M. van Sinderen. Flexible home care automation adapting to the personal and evolving

needs and situations of the patient, 2009.

[23] Department of Health. Our health, our care, our say: a new direction for community services. Product number 270875, 2006.

[24] G. Fischer, R. McCall, and A. Morch. Design environments for constructive and argumentative design. In *Proceedings of Computer Human Interaction (CHI) conference on Human factors in computing systems, ACM SIGCHI*, pages 269–275, Austin, Texas, 1989. ACM.

[25] B. N. Schilit and M. M. Theimer. Disseminating active map information to mobile hosts. *IEEE Network*, 8(5):22–32, 1994.

[26] V. Bellotti and K. Edwards. Intelligibility and Accountability: Human Considerations in Context-Aware Systems. *Human Computer Interaction*, 16:193–212, 2001.

[27] P. Dourish. Developing a Reflective Model of Collaborative Systems. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 2(1):40–63, 1995.

[28] A. MacLean, K. Carter, L. Lovstrand, and T. Moran. User-Tailorable systems: Pressing the Issues with Buttons. In *Proceedings of Computer Human Interaction (CHI) conference on Human factors in computing systems, ACM SIGCHI*, pages 175–182, Seattle, Washington,USA, 1990. ACM.

[29] S. Fickas. Clinical Requirements Engineering. In *Proceedings of the 27th International Conference on Software engineering (ICSE)*, pages 140–147, St. Louis, Missouri, USA, 2005. ACM.

[30] J. O. Kephart and D. M. Chess. The vision of autonomic computing. *Computer Magazine*, 36(1):41–50, 2003.

[31] C. Boutilier, R. Das, J. O. Kephart, G. Tesauro, and W. E. Walsh. Cooperative negotiation in autonomic systems using incremental utility elicitation. In *Uncertainty in Artificial Intelligence*, pages 89–97. Morgan Kaufmann Publishers Inc., 2003.

[32] J. S. Clark and M. McGee-Lennon. A Stakeholder Centered Exploration of the Current Barriers to the Uptake of Home Care Technology in the UK. Technical Report TR-2009-314, Department of Computing Science, University of Glasgow, 2009.

[33] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[34] T. McBryan. *A Generic Approach to the Evolution of Interaction in Ubiquitous Systems* . PhD thesis, University of Glasgow, 2011.

[35] D. Marples and P. Kriens. The Open Services Gateway Initiative: An Introductory Overview. *Communications Magazine, IEEE*, 39(12):110–114, 2001.

[36] K. Turner, S. Reiff-Marganiec, L. Blair, J. Pang, T. Gray, P. Perry, and J. Ireland. Policy Support for Call Control. *Computer Standards and Interfaces*, 28(6):635–649, 2006.

[37] F. Wang, L. S. Docherty, K. J. Turner, M. Kolberg, and E. H. Magill. Service and Policies for Care at Home. In *International Conference on Pervasive Computing Technologies for Healthcare*, Innsbruck, Austria, 2006. IEEE Computer Society Press.