Towards an Intuitive Interface for Tailored Service Compositions

Steffen Higel, Tony O'Donnell, Dave Lewis, and Vincent Wade

Knowledge and Data Engineering Group
Department of Computer Science
Trinity College Dublin, Ireland
{Steffen.Higel,Tony.ODonnell,Dave.Lewis,Vincent.Wade}@cs.tcd.ie

Abstract. Novel modes of interaction and tailored application delivery are a key challenge in ubiquitous computing. This paper proposes a method of adaptively delivering lightweight, disposable service compositions to a user in a smart space. The user intuitively requests these services through a single, environment interface, which infers goals through observing behaviour and context, and manages the configuration and interaction of the devices within the space. The composition is driven by generating tasks from these inferences, which can be resolved down to candidate services. We will discuss the proposed interaction between these components and highlight what direction our research will take.

1 Introduction

The human-computer relationship has moved from single machines with multiple users, through the PC age and on to spaces where many users are surrounded by a myriad of computing devices [Vertegaal, 2003]. In such an environment, the user cannot possibly interact with every distinct device, let alone every piece of software on these devices, and so instead we propose an interface to the environment which monitors a users actions and then communicates their needs to web-service representations of the functionality on those devices in the space. For developers, this emerging scenario introduces many challenges from efficiently building this interface, to delivering useful support. Our work proposes a system which will address this challenge. This paper outlines our research and highlights how natural interfaces can be used to enable users to interact with a ubiquitous computing environment, and also how support can be delivered through the dynamic composition of tailored, lightweight, disposable services.

2 Background

The research discussed in this paper divides into two areas, namely developing the environment interface and then providing support to users accessing it. Presented below is a brief background to these.

J.-B. Stefani, I. Demeure, and D. Hagimont (Eds.): DAIS 2003, LNCS 2893, pp. 266–273, 2003.

2.1 Adaptive Techniques

Software in its traditional form has been driven predominantly by a fixed set of design decisions, defined before development begins, which place finite limits on the capabilities of the end product. Although research into software which can learn and adapt to the preferences, behaviour and context of the user[Langley, 1997] has existed for quite some time, it was only during the 1990s that software applying these techniques to commercial software like AVANTI and indeed it is only in recent years that we have seen any definite move in the direction of products being used by end-users which feature adaptive components[Horvitz, 1999]. These agents can monitor a user's interaction with software and take actions on behalf of the user based on these assumptions.

Although in adaptive hypermedia systems, an area in which we have conducted a large amount of research, metadata (the data which describes the data) driven techniques made some headway in describing the scope of a piece of content, it did not operate on a sufficiently high level to give any meaningful hints to the system itself as to what goals this piece of content was trying to fulfill[Dagger et al, 2003]. The navigation through a collection of hypermedia documents is abstracted such that it can be viewed as a navigation through a series of concepts represented as nodes. These concepts will have one or more pieces of appropriate candidate pieces of content associated with them. This idea of candidacy can effectively be applied to another area of adaptive software, as will be demonstrated in this paper. The idea remains sound if you use it to abstract the functionality of an email program (the tasks like "view list of emails", "read specific email" and "write email" can be abstracted from the software libraries that provide them) or in service composition (where the functionality provided by a service can be viewed as being distinct from an instance of this service).

2.2 Ubiquitous Interfaces

With the ever-decreasing size and increasing power of computers embedded processors are appearing in devices all around us. As a result the notion of a computer as a distinct device is being replaced with a ubiquitous ambient computing presence[Dey et al, 2001], e.g. wearable computers. This proliferation of devices will present user interface designers with a challenge. While an average user might cope with having a different interface for their PDA, desktop PC and mobile phone, they will certainly have difficulty if the range of devices is greatly increased. In the past, designers have suggested creating a single interface appearing on all devices, however research has thus far not proved this to be the optimum solution[1]. We instead propose an ambient environment interface within the computing environment which observes the users activities and then acts on what the user wants. The environment then handles the individual interaction

[1] For example, developers of the Symbian OS found it was not feasible to offer the same user interface on Symbian-powered PDA's as on desktop computers.

with the devices, customising and configuring them, thus sparing the user. Later in this paper we will more fully describe such an environmental interface technology.

2.3 Service Composition

The field of service composition covers the techniques used to bind together two or more software components capable of intercommunication. Recent interest in the topic has surged due to the emergence of web-services - applications running on the World Wide Web, which use standardised methods for communication and self description[Edgar, 2001]. By dynamically chaining a number of these services together at run-time[Kcman et al, 2001], functionality and convenience can be added to the services to which an end user might traditionally be accustomed - those offered by a single vendor accessed with a web browser[Wu et al, 2003].

3 Proposed Architecture

Presented are the designs of two systems, an environment interface architecture and an automated and tailored service composition architecture, providing sufficient background to understand how an interaction between the two might take place.

3.1 TSUNAMI

TSUNAMI, Tailored Support of Users' Natural Activities with Mixed Initiative, is an interface technology that we are currently researching. The system monitors users for implicit inputs, such as vague gestures or conversation, and explicit inputs, such as verbal or typed commands, and uses these to predict what assistance the user requires to fulfill their perceived goal. Predictions are also guided by context information such as calendars, location and biographical information. Support is delivered with mixed initiative[Horvitz, 1999], that is, the system adjusts the pre-emptive support it gives on a per-user and per-task basis and is aimed at reducing the cognitive load placed on the user.

Once a confident prediction is made, a task request is formed and dispatched to the Adaptive Service Composition Engine, ASCE, so that a solution can be prepared. However, at all times the user maintains overall control and can disregard the assistance being offered.

Inputs. TSUNAMI accepts inputs from a range of sources. Some of these allow natural actions to be used by means of sensors in the space. These will supply data on physical behaviour such as manipulation of objects, gestures or speech. Calendars can add temporal context. A user's biographical information helps inform predictions by indicating prior behaviour and preferences. The business\work context can ensure that predictions do not contradict corporate policies or assist with anti-social activities. Finally, the space can be equipped with declarative tools such as whiteboards or traditional PC-like interfaces. These allow unambiguous input.

In order to make these inputs intelligible to the system, they are first parsed to extract meaning. This parsing will use pattern matching based on existing cached knowledge. Where no pattern exists, or when parsing proves unsuccessful, some other remedial action will have to be taken. The nature of this action has yet to be investigated.

Prediction Selection. Each of the inputs above has a certain level of ambiguity ranging from the relatively clear declarative inputs to the much less precise sensor data. In order to reflect this, different types of inputs come with varying levels of confidence as to what they are trying to communicate. For example, if user Kate enters the space and says ‘print myfile.doc’ the system can be confident that it should do just that. On the other hand, if she makes a comment about the weather it may not mean that the air-conditioning should adjust itself.

TSUNAMI will address this problem in two ways. Inputs will have associated confidence levels, and combinations of inputs will produce aggregate confidence[2]. Given tasks will have minimum levels of confidence associated with them. This task confidence will be similar to that expressed in [Horvitz et al, 1998], where a measure of the utility of offering assistance successfully or otherwise, rather than not offering help, can be measured and used to arrive a point where the system can pre-emptively offer support.

In addition, TSUNAMI will provide a mechanism for evidence to degrade appropriately over time, so that a comment made an hour ago does not overly inform a prediction being considered now. It will also allow predictions to similarly decay, and introduce a lower threshold after which they will no longer be considered.

TSUNAMI will maintain a number of competing prediction hypotheses and once the aggregate confidence for a given prediction has been exceeded a meaningful request can be sent to the ASCE for a solution. In certain cases, where either the user is unhappy for the system to be pre-emptive or where the task is mission critical, TSUNAMI may interrupt the user for confirmation before sending this request.

3.2 Adaptive Service Composition

People have varying preferences regarding the behaviour and presentation of software. We have no reason to believe that composite services should be any different. A potential key advantage of dynamically composed software over monolithic ones is the ability for a given end user to at least express their preferences in these areas, if not to modify a service composition themselves. If web-service

[2] It is expected that input aggregation will make use of complex event processing [Luckham, 2002] techniques to allow simple events to be combined to form more meaningful higher-level composite events - e.g. making a phone call rather than lifting a receiver, finding a number, dialling and then speaking.

wrappers for existing software are developed, it is possible to very quickly create composite services with functionality that would take a considerable amount of time to develop using a monolithic model.

System Overview. ASCE, the Adaptive Service Composition Engine is a proposed system to facilitate the semi-automated creation of service compositions tailored to a user's needs[Conlan et al, 2003]. This approach to service composition has the potential to provide a series of benefits to software developers and end users alike. From the developer's and advanced user's perspective, the speedy and semi-automated selection and linking of distributed components saves time and effort. From the end user's perspective, this approach could result in applications which are coupled more closely with the their needs and preferences, along with providing a piece of software which offers levels of functionality greater than that of a monolithic service.

Details of Operation. The engine abstracts the subtasks that are needed to complete a given goal from the actual services that can fulfil them; as previously discussed, the functionality provided by a service is considered distinct from an actual instance of a service which provides that functionality. The Service Composition Toolkit is tasked with the former while the Service Integration Engine handles the latter. The process of adaptive service composition begins with the system receiving a description of a task (the format of which will be discussed later), which is passed to the Service Composition Toolkit. A cache of previously decomposed tasks is searched for similar initial requirements.

Assuming none are found, the toolkit will attempt to decompose the task itself, using service classifications (descriptions of what kinds of service exist, i.e. pay for something using a credit card, recommend a book, provide content on a specific topic to a student) down to a set of rules defining the required services and the order in which they should react. The service composition toolkit should be able to fall back on the input of a domain expert, a human with sufficient skill to break the task down using information provided by the service classifications, should the semantic gap between service descriptions prove to be too wide.

The Service Integration Engine takes the list of service classifications and maps them onto available services. It is assumed that there may be many services available which provide similar functionality. Each of these candidate services should be compared with the user profile, the user context and quality of service requirements, resulting in the most appropriate candidate being selected.

4 Integration and Inter-operation

The concept of service composition can ultimately facilitate the dynamic generation of useful composed services by an end-user who has received only a small amount of training. When this is combined with an intuitive interface there is a significant potential for user empowerment. However, to be truly effective, there must be efficient and well-structured communication between the two systems. Despite being discrete areas of research, which in the future will function independently but are still capable of inter-operation, it is still worthwhile for the systems to be examined from the outside-in. This will help to define the responsibilities and to define what data should be accepted and returned by of each system.

At the core of any interaction between TSUNAMI and the ASCE is the information passed them. This data will describe the tasks that TSUNAMI has interpreted from the actions of a user. The challenge is finding a useful balance between preparing a heavily detailed description of a task (which would expect too much of the TSUNAMI and too little of the ASCE) or sending a vague, light task description. It would seem reasonable to assume from this that the ASCE must be presented with some structured description of the task, with much of the contextual interpretation already performed by TSUNAMI.

It is therefore proposed that the task description passed from TSUNAMI to ASCE be expressed unambiguously in terms of inputs, pre-conditions, effects and outputs, rather than a vague, verbose specification. The generation and interpretation of these conditions, effects and outputs are still complex, but they place reasonable, structured, intelligible and logical requirements on each component.

Figure 1. Shows the inter-operation of the two systems, though its workings are best illustrated with a practical example:

Dave is on his way from his own company's office building to another regional headquarters. He is making this trip by car, and has just realised that he has forgotten the location of the meeting room and the exact time of the meeting. He also wants to check over the minutes of their previous meeting. His PDA hears him say "I need the email from Jim which details the meeting, the meeting location r and also has the minutes of the previous one." His PDA, acting as a sensor for TSUNAMI interprets this and passes its description of Dave's task (get the email from Jim about the meeting). TSUNAMI uses contextual information to resolve this down to a set of inputs, outputs, preconditions and effects:

Inputs: `Dave, need email from Jim with meeting details.`
Outputs: `Email and minutes displayed on appropriate device.`
Effects: `Meeting time and location are placed in Dave's diary.`
Preconditions: `None.`

The ASCE has access to services which provide information on Dave, including the URI to his email Inbox, a service which can access such a mailbox and search through it and another which can connect to some centralised diary system like Microsoft Exchange or Lotus Notes and add entries to it. It will compose these services together, searching a path through it's available services to get from the required inputs to the outputs and effects. Finally, the task of displaying the results must be handled. The ASCE has access to the contextual information which places Dave in his car on the way to the regional headquarters. It creates a new service composition which will convert and filter the document and then send it to the printer in the lobby of the headquarters. These composi-

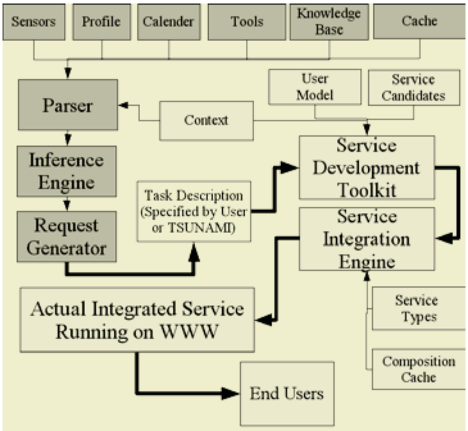

Fig. 1. The architectures of both the TSUNAMI and ASCE systems, with the common link between them.

tions are executed, the web services performing the tasks required of them and all of the information Dave requires is waiting for him when he arrives.

5 Conclusion and Further Research

It is the belief of the authors that a novel approach to user support in a ubiquitous computing environment has been proposed. The methods used to enable co-operation between existing techniques in adaptive composition and human computer interaction have been highlighted, along with future research directions of these areas. In addition, the research presented has a clearly defined focus, with the potential for experimental tests in the near future.

To assist with the design of TSUNAMI, a review of relevant research was conducted, including human-computer interaction, event processing, mixed- initiative support\attentive user interfaces and natural interfaces[3]. At present, a TSUNAMI testbed is under development which will simulate simple user interaction with the system. This simulation will allow a user to interact with a 2-dimensional model of an office, and will monitor point of view and speech. This experiment will test the basic system assertions, namely that our chosen inputs can be recorded, parsed, used to build complex event patterns and that these can be used to trigger task requests. Once the basic assertions have been proven, a more comprehensive test programme will begin. This will see an increased range of input axes, devices and deliverable tasks. Ultimately, it is envisaged that a full real world test will be carried out.

Research in the area of service composition has focused on two areas; developing services which can be used in a composition and the design of the Adaptive

[3] See assorted documents at `http://www.cs.tcd.ie/Tony.ODonnell/publications/index.html`.

Service Composition Engine. As a testbed, a series of file conversion services will be built atop the traditional tools found on UNIX operating systems. Because the path used to convert one file format to another may not be trivial, it will provide an interesting challenge for the ASCE to see if it can automatically resolve the high level tasks down to actual services. At present, a basic tree search of DAML-S documents is being developed along with an investigation into existing automatic service composition techniques. Once this initial series of tests has been completed, work will begin on defining methods used for describing service compositions to aid storage and re-use along with integration with parallel research into quality of service and access control.

This work is funded in part by the Higher Education Authority (HEA) of Ireland under the auspices of the M-Zones project[4].

References

Edgar, 2001. Web Services: A Position Paper, Edgar, G., W3C workshop on Web services, 2001

Dey et al, 2001. References Distributed and Disappearing User Interfaces in Ubiquitous Computing, Dey, A., Ljungstrand, P., Schmidt, A., 2001

Langley, 1997. Machine learning for adaptive user interfaces, Langley, P., German Annual Conference on Artificial Intelligence, 1997

Kcman et al, 2001. Towards Zero-Code Service Composition, Kcman E., Melloul L., Fox A., Hot Topics in Operating Systems, 2001

Wu et al, 2003. Automatic Web Services Composition Using SHOP2, Wu D., Sirin E., Hendler J., Nau D.,Twelfth World Wide Web Conference, 2003

Conlan et al, 2003. Applying Adaptive Hypermedia Techniques to Semantic Web-Service Composition, Conlan O., Lewis D., Higel S., O'Sullivan D., Wade V., Adaptive Hypermedia 2003

Horvitz, 1999. Principles of Mixed-Initiative User Interfaces, Horvitz, Eric, Proceedings of the SIGCHI conference on Human factors in computing systems, pages 159-166, 1999

Dagger et al, 2003. An Architecture for Candidacy in Adaptive eLearning Systems to Facilitate the Reuse of Learning Resources, Dagger D., Conlan O., Wade V., eLearn 2003

Horvitz et al, 1998. The Lumiere Project: Bayesian User Modeling for Inferring the Goals and Needs of Software Users, Horvitz E., Breese J., Heckerman D., Hovel D. & Rommelse K., Fourteenth Conference on Uncertainty in Artificial Intelligence Intelligence, 1998

Vertegaal, 2003. Attentive User Interfaces, Vertegaal R., Communications of the ACM, Vol. 46, N0. 3, pages 31 - 33, March 2003

Luckham, 2002. The Power of Events: An Introduction to Complex Event Processing, Luckham, David, Pearson Education Inc, 2002

[4] [M-Zones] M-Zones (Managed Zones) `http://www.m-zones.org`.