

Towards Context Oriented Web Services for Smart Personal Object Technologies (COWSPOTS)

David Dahlem, Yury Bychkov, Luay Kawasme, Jens H. Jahnke

UNIVERSITY OF VICTORIA

Department of Computer Science

Victoria, B.C, V8W 3P6, Canada

Voice: +1 (250) 472 4542, Fax: +1 (250) 721 7292

[ddahlem|ybychkov|lkawasme|jens@cs.uvic.ca]

Abstract

Research in the area of Context-Awareness has traditionally focused on location, time, and proximity-based context sensing. Advances in wireless networking technology and in the miniaturization of computing hardware now allow us to contemplate more expansive utilizations of contextual information. In this paper, we advocate the use of Web services as the basis for the next generation of context aware applications. Web services potentially offer unprecedented cross-platform interoperability and can be used to broaden the range of context indicators and services for mobile devices. Also in this paper, we introduce the COWSPOTS project, a research initiative that seeks to apply context-aware Web services to the medical domain.

1. Introduction

Context-enabled applications offer many advantages to the mobile user. Specifically, the use of environmental information to modify program behavior allows for a more dynamic and user-friendly system, thereby, enhancing interactions between human and computer. Mobile users are especially aided by context-enabled applications because their physical environment is ever-changing. Moreover, context-enabled applications, which offer the potential to reduce the amount of user interaction with the device, are particularly important for mobile users because mobile devices typically possess impoverished user-input capabilities.

Although context-enabled applications offer potential rewards for mobile users, widespread use of such systems has not yet been realized. To date, most research in the area of context awareness has produced systems that are tied to proprietary technology and, therefore, have circumscribed mainstream utility. For this reason, we advocate the use of Web services as the basis of the next generation of context aware applications. Web services potentially offer unprecedented cross-platform interoperability, an important asset considering the extreme diversity of mobile device platforms. Further, Web services can be used to broaden the range of inputs used by context engines to infer context.

In this paper, we introduce a new research initiative called Context Oriented Web services for Smart Personal Object Technologies (COWSPOTS). COWSPOTS, currently in the early stages of development, will utilize Web service-based context-awareness to provide enriched services to medical professionals. We will provide a brief description of our proposed COWSPOTS research in section 4. The rest of the paper is structured as follows. In section 2 we define context awareness in the scope of related research. In section 3, we discuss how Web services, when applied to context-enabled systems, can be advantageously used for enriching context inferences and service delivery. In section 5, we provide a COWSPOTS user scenario to illustrate the utility

of Web service-based context-awareness within the medical domain. We conclude with a summary of the key points argued in this paper.

2. Context Awareness and Related Work

Research in the area of context-awareness is relatively well-established, with roots reaching back to the early 1990's. Perhaps the earliest context aware system was the Active Badge Location System [7], used for tracking the location of people and equipment within an organization. Until recently, most research in context awareness conducted has focused primarily on location, time, and proximity-based contextual factors. A survey of major context aware research initiatives [1] reveals that the emphasis of context aware systems implemented before the year 2000 utilize location and time as the primary sources of context determination.

With the focus of most early context aware systems on physical sensor-based context factors, especially location and proximity, Schilit and Theimer [5] defined context-aware computing as software that "adapts according to its location of use, the collection of nearby people and objects, as well as changes to those objects over time." However, recent advances in wireless networking and the miniaturization of computer hardware have provided the basis for providing a more general definition of context and a widening scope of context-enabled applications. Early acceptance of the Schilit and Theimer definition of context-aware computing has been largely replaced by the more general definition offered by Dey [3]:

A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task.

Mobile devices now have better access to distributed resources and therefore can access contextual information not strictly in the 'line-of-site' of the device.

Germane to the topic of context-aware Web services, some researchers are exploring the use of XML and Web services in delivering context-enabled applications. The WASP project [8] aims to develop a Web services-based context-aware service platform built upon 3G networks. The SCARLET project as described in [6] is another Web service-based context-aware system in which Web services are explicitly utilized to support cross-platform communications and to promote extensibility of the system. Inclusion of Web service standards in emerging context-aware applications indicate that the user context need not be confined to immediate surroundings of the mobile user.

3. Web Services for Context-Aware Applications

XML Web services (or just simply, Web services) is a relatively new programming paradigm that leverages the universality of XML to promote communications between heterogeneous systems. As stated previously, platform heterogeneity and diversity is especially acute in the realm of mobile computing (and pervasive computing in general). Web services help to obscure incompatibilities between operating systems (e.g., Pocket PC, Palm), programming environments (e.g., J2ME vs. .NET Compact Framework [2]), and mobile device types (e.g., wrist mounted watch, PDA, mobile phones). Moreover, because Web services are delivered over TCP/IP networks, the various wireless network infrastructure types utilized by mobile devices (e.g., 802.11b, Bluetooth, GSM, etc.) can be abstracted from the context-aware system.

In addition to improving cross-platform interoperability, Web services offer a standardized way to access an ever-growing range of services and information sources. This offers two main advantages. First, a Web service-sourced context inference engine has more context indicating information sources from which to determine user context. Some or all of these indicators may be derived from sources external to an organization, such as a weather or traffic Web service; others could be derived from intra-organizational sources like an accounting department or food services web service. Second, mobile users have more options in obtaining services and

information for given contexts. For example, a mobile user at a bus stop can receive the latest bus schedule from a city transportation web service.

However, Web services do come with a significant performance penalty, as reported in [2]. Few personal digital assistants (PDAs), and even fewer mobile phones, possess the resources to properly handle Web services. Implementations are constrained by the imposed requirement of parsing XML documents, which consumes physical resources, like memory. In addition, text based XML documents require larger bandwidth for transmission compared to binary data. However, as it is the case for all technologies, we believe that PDAs with more affluent resources will become more affordable, which will overcome the memory and bandwidth constraints.

4. COWSPOTS

4.1. Overview

In order to test our ideas about the use of Web services in context-aware mobile computing, our research group started developing a project called COWSPOTS (Contextual Oriented Web services for Smart Personal Object Technologies). General usage scenario of the COWSPOTS enabled system can be informally described as follows:

1. User picks up a mobile device and logs in to the system.
2. System monitors various parameters (both physical (e.g. time, location) and virtual (e.g. schedule, to-do list, etc.)) and determines when user's context changes.
3. System re-evaluates which contexts should be currently active and determines a set of applications (so called SPOTlets) that user needs.
4. Appropriate SPOTlets (along with the necessary data) are downloaded (or activated, if they are already available) to the user's mobile device and initialized with the required context-dependant data.

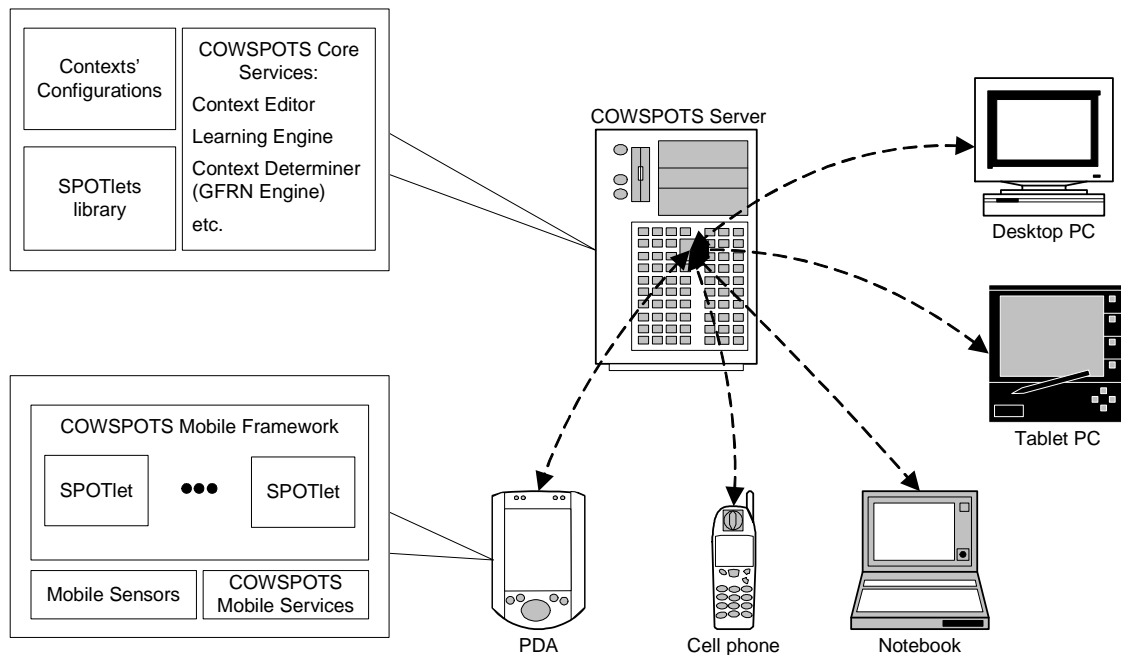


Figure 1. COWSPOTS' architecture.

4.2. Architecture

COWSPOTS' infrastructure consists of a server and multiple clients (Figure 1). Any type of computing device can potentially serve as a COWSPOTS client if it is connected to the Internet. Most of the clients would be mobile (e.g. PDAs, cell phones, tablet PCs), however some users might want to use COWSPOTS on their desktop PCs as well.

The main purpose of the client is to allow the user to interact with a context-dependent set of SPOTlets, though it also has to perform the necessary overhead functions that are required for context determination and SPOTlets management. Since the capabilities (both memory and performance) of the mobile devices are still relatively weak (even though there have been a considerable progress in the last couple of years), most of the overhead processing is done on the server and the clients only have to host some remote sensors and accept downloads of SPOTlets and data.

COWSPOTS server has four main functions:

- Stores the contexts' configurations for all users of the system.
- Contains the library of SPOTlets and parts of the data for them (only what the users decided to host on it).
- Performs most of the overhead processing that is required by the system: evaluates the context changes, sends required SPOTlets, synchronizes data, etc.
- Allows the users to setup and edit context configurations (both SPOTlet sets and activation conditions)

Now that we have given the overview of how the COWSPOTS enabled system works, we can address some finer points.

4.3. Activation Conditions

One of the most important issues is "how can we determine when a context is active". Our proposed solution for that is to have a possibilistic distance metric called **Activation Condition (AC)** associated with each context. In order to make ACs both powerful and flexible while making sure that they are not expensive to compute, we have decided to define them using Generic Fuzzy Reasoning Networks (GFRNs). GFRNs are graphical networks of predicates that are connected by heuristic rules modeled as possibilistic implications with associated confidence measures (more details can be found in [4]). GFRNs are based on a well-developed formal foundation and their use, among other things, will allow us to reason about various properties of contexts.

The ACs are defined by the user (directly or via the learning mechanism [see Section 4.4]) and are based on a variety of parameters that encompass the user's idea about what a particular context should be (so in a way a context is actually defined by its AC). Some of the parameters are external to the user (e.g. time, location, etc.) and some are internal (e.g. schedule). There is no limit on the number of parameters as well as on what can be used as a parameter. Though the values for some of the parameters are read from the physical devices (e.g. hardware clock, proximity sensors, etc.), as mentioned in section 3, all of them are wrapped in the Web services. These wrappers could be very simple (sending just the read value) or arbitrarily complex (and likely more useful). For example, a GPS-based location sensor might be wrapped in the Web Service that will express user's location not in terms of latitude and longitude, but as a distance from hospital, or, if it can measure speed, as an approximate travel time.

As mentioned before, the server obtains all parameters from various Web services and periodically re-evaluates Activation Conditions for the user's contexts. When the AC's value rises above (or falls below) some predefined threshold, the corresponding context becomes active/inactive.

Note: in the future version the thresholds might become adaptable to allow even greater flexibility.

4.4. Learning

Another important issue is the initial configuration of the system. For a new user of the system, manual creation of all contexts that he/she wants to have (even using a good editor) is going to be slow and tedious, thus the better solution would be to employ a learning mechanism. As we envision it now, the learning process would consist of the following three phases:

1. **Contexts gathering phase**

This phase is very simple and is designed to gather as much information about contexts as possible. The user manually adds the SPOTlets that he/she would like to use at a particular moment and removes useless ones. Whenever the user thinks that current state is worthy of being a context, he/she clicks a “Save” button and the SPOTlets configuration as well as the values of all possible parameters (since at that moment we don’t know which parameters are relevant) are saved.

2. **Contexts simplification phase**

During the learning phase every time the user clicks “Save” button, a new context configuration is created, so it will most likely result in a set of contexts that is far from optimal. The goal of the second phase is to simplify it using various algorithms. Their detailed descriptions are beyond the scope of this paper, but, for example, if two contexts have the same SPOTlets configurations, then they can be joined into one (and their AC’s are combined appropriately).

3. **Contexts simplification phase**

This phase is similar to the simplification phase, only instead of reducing the number of contexts, we are simplifying the contexts themselves. E.g. if one context is a sub-context of another (i.e. it’s always active whenever it’s super-context is active) then we can remove their common SPOTlets from the super-context.

5. A Day in the Life of a COWSPOTS User

To illustrate our vision of how COWSPOTS will deliver context-sensitive Web services to medical professionals, we provide an example user scenario (see Table 1). Dr. Margaret Smith, a fictional orthopedic surgeon, has access to a variety of COWSPOTS-enabled devices in the hospital and in her home. This scenario illustrates the following goals of COWSPOTS:

- Availability of different services in different contexts without being bound with the device’s capabilities.
- Context information is established from multiple sources. A subset of these sources exists on the COWSPOTS user side, e.g. location. Other contextual sources are acquired from online services, e.g. Online Calendar.








TIME	SCENARIO	COWSPOTS DEVICES	Relevant Context Indicators	ACTIVE SPOTLETS
9:00 – 9:30	Dr. Smith's first task in the morning is to examine the condition of her in-patients (both pre-operation and post-operation). She will want to see the latest laboratory information of her pre-ops, and will want to observe overnight progress of her post-ops by reading reports submitted by night-duty nurses.		Online schedule service Location: In hospital	Patient history viewer Lab data viewer Patient hospital-stay viewer
9:30 – 10:30	Dr. Smith meets with each post-op patient and, among other things, evaluates whether the current level of medication is appropriate. She utilizes a decision-support service (DSS) for determining if a new dosage of a certain medication falls within established guidelines and if such a change will adversely react with other medications		Online schedule service Proximity to individual patients	Patient history viewer Lab data viewer Patient hospital-stay viewer Medication DSS
10:30 – 12:00	Dr. Smith participates in a weekly meeting with other hospital personnel (physicians, nurses) to discuss hospital procedures and protocols. Each meeting participant receives on their COWSPOTS-enabled device a meeting agenda and minutes from the last meeting. Since this is an important meeting, Dr. Smith does not want to be interrupted by non-emergency alerts on her pager or PDA and therefore wants them transferred to her answering service.		Online schedule service Proximity to meeting participants Location: Meeting room	Meeting organizer (PDA) Smart alerting service (PDA and Mobile Phone)
12:00 – 13:00	At meeting's end, Dr. Smith is ready for lunch. She obtains the latest menu and the size of the line-up in the hospital cafeteria. Over lunch, she wants to view a list of non-emergency alerts (either e-mail or phone-based) that were diverted to her answering service during the meeting.		Online schedule service Location: Cafeteria	Cafeteria online viewer Answering service viewer
13:00 – 17:00	Dr. Smith is in surgery for the next three hours. She does not want to receive any alerts (emergency or otherwise) while in this mode; all alerts will be handled by an answering service		Online schedule service Location: Operating room	Smart alerting service
17:00 – 17:30	Dr. Smith exits the hospital and embarks on her journey home. She wants to receive the latest traffic information.		Device type: 3G phone Location: Out of hospital	Traffic information service viewer
17:30 – Next Day	Dr. Smith is at home. She wants to be notified if the condition of any of her in-patients deteriorates significantly. Otherwise, all non-emergency calls are transferred to an answering service. If an emergency situation arises, she wants access to patient information and to other services such as the medication decision support system.		Online schedule service Location: Out of hospital	Patient history viewer Lab data viewer Patient hospital-stay viewer Medication decision-support system

Table 1: A COWSPOTS User Scenario

6. Summary and Future Work

In this paper, we argued that context-aware systems can be advantageously augmented with Web service technology. As mobile technology improves, and as our collective notion of context becomes more expansive, Web services provide the means by which to deliver an ever-broadening range of services, thereby enriching context-enabled applications. We introduced a new research initiative COWSPOTS that we propose will take advantage of Web service technologies to enhance context-awareness.

We are exploring a few research avenues for COWSPOTS, including:

- Using context indicators, such as location, time, fingerprint provided, password entered, etc., to manage access to sensitive data and to provide a more secure, yet user-friendly system
- Developing a formal GFRN-based foundation for contexts that will allow us to reason about their properties (which is especially important for security-related contexts).

7. References

- [1] G. Chen and D. Kotz. A Survey of Context-Aware Mobile Computing Research. Technical Report TR2000-381, Computer Science Department, Dartmouth College, Hanover, New Hampshire, November 2000.
- [2] D. Dahlem, J. Jahnke, A. Onabajo, and Q. Wang. The Challenges of Implementing Web Services on Small Devices. In *Proceedings of the Workshop on Pervasive Computing, Going Beyond the Internet for Small Screens, OOPSLA 2002*, Seattle Washington, November 4, 2002.
- [3] A. Dey. Understanding and Using Context, *Personal and Ubiquitous Computing (2001)* 5:4-7.
- [4] J.-H. Jahnke, W. Schafer, and A. Zundorf. Generic Fuzzy Reasoning Nets as a basis for reverse engineering relational database applications. European Software Engineering Conference (ESEC97), Zuerich. Springer, LNCS 1301.
- [5] B. Schilit and M. Theimer. Disseminating Active Map Information to Mobile hosts. *IEEE Network*, 8(5):22-32. 1994.
- [6] P. Wagstrom, Scarlet: A Framework for Context Aware Computing. Unpublished Master's Thesis. Illinois Institute of Technology. July 2003.
- [7] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The Active Badge location system. *ACM Transactions on Information Systems*, 10(1):91-102, January 1992.
- [8] M. Zuidweg et al. Using P3P in a web services-based context-aware application platform. Web resource: <http://www.w3.org/2003/p3p-ws/pp/utwente.pdf>