A Context-Sensitive Rule-based Architecture for a Smart Building Environment

John Herbert, John O'Donoghue and Xiang Chen
Department of Computer Science, University College Cork, Ireland.
E-mail: { j.herbert, j.odonoghue, cx1}@cs.ucc.ie

Abstract

In a smart building environment nomadic users can benefit from specialised context-sensitive services. These can increase productivity and offer an improved lifestyle for users. The Data Management System-Context Architecture (DMS-CA) has been designed to provide these services. The DMS-CA is novel in providing a completely generic system that is both user-friendly and efficient. The system is generic in that it can be tailored by the system administrator (using XML specifications) for a particular environment of context information inputs and a set of services. In addition, the actual rules that determine which services are triggered by which combination of contextual inputs are determined by the individual user in a user-friendly manner. Thus the generic architecture can be easily deployed for a particular building and an individual user. Generic solutions are sometimes inefficient but this system has been designed to be highly efficient through using an event-based architecture and dealing with time in a special way. The DMS-CA has been deployed for a building incorporating a wireless sensor network, and shown to provide a high quality, efficient context-sensitive data delivery service.

1. Introduction

The Data Management System-Context Architecture (DMS-CA) has been designed to provide context-sensitive services in a smart building environment. This forms a specialised part of a general Data Management System (DMS) [1,2,3] that manages data in a wireless network based pervasive environment.

The target smart building application uses a set of wireless environmental sensors based on the Tyndall Mote (cf. [2]). This is a multilayer mote incorporating pluggable sensor layers. It can support many different kinds of sensors such as temperature, light level detection and humidity, and it has previously been used in various applications including medical and environmental.

A variety of implementations of context-aware applications have been reported, e.g. [4,5,6,7,8,9,10,11]. An example of a typical application is the Context-Aware Personalised Service Delivery [6] that provides an intelligent multi-agent system for context-aware service delivery and recommendations to mobile users, As well as different applications, solutions such as The Hydrogen Approach [10] address the issue of having an appropriate architecture to deal with context.

The approach presented here tackles similar issues in the context of a smart building environment. The main focus has been different, however. The goal has been to produce an architecture with a novel set of attributes, namely, that the solution should exhibit generality, efficiency and user transparency. This paper presents an overview of the DMS-CA and how it achieves these goals.

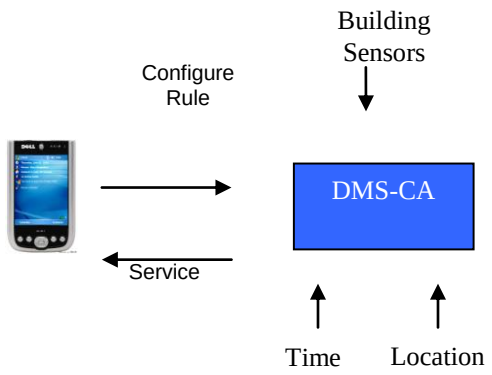

Figure 1. Overview of the Data Management System – Context Architecture (DMS-CA).

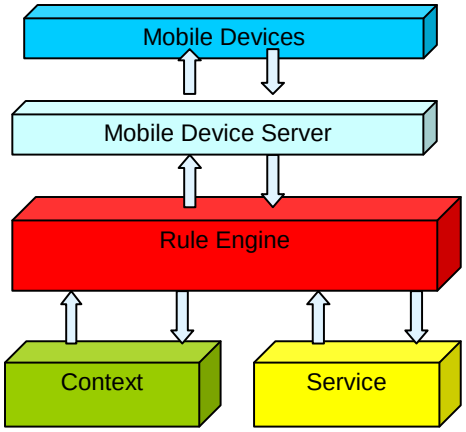

Figure 2. System Architecture

2. Architecture

The DMS-CA takes input from various building sensors, and contextual information such as the current user profile, user location, user schedule and current time. The DMS-CA then executes the contextual rules and when the current context triggers a certain rule then the service specified by that rule is executed. Thus a rule might specify that if a user enters the meeting room between 3pm and 5pm then a certain file is downloaded to the user's mobile device. Another example might be a rule that is triggered when the temperature in an office falls below 22C on a weekday between 8.30am and 5pm, and this results in a message being sent to the appropriate activator to heat the office.

Overviews of the DMS-CA and its architecture are given in figures 1 and 2. As shown in figure 2, the architecture comprises of four layers where the lowest layer contains the actual contexts and services that are registered for a particular application and the other layers deal with the definition and execution of the rules that relate services to contexts. Each layer is just concerned with the layer it interfaces to. For example, mobile devices do not need to know how to control the rule engine nor how to invoke services, and the server does not need to know how often the rule engine will check the context.

The Mobile Devices layer contains all the mobile devices interacting with the server, so it is also in effect the client layer. It sends user defined rules to the Mobile Device Server, and gets acknowledgement of the service offered when conditions in the rule hold. The Mobile Device Server layer is like a controller that interacts with distributed clients. This layer is in charge of offering the options the users can use to define rules, and of accepting users' rules and injecting them into the rule engine. Once conditions in a rule hold it sends commands to the rule owner's mobile device to take the specified action. The Rule Engine layer checks the context according to all the users' rules, and offers services to users once the rule conditions hold. The context and services are placed

on the same layer because their structures and management mechanisms are similar, and because they are governed by similar conditional clauses. They are both managed by the rule engine.

The main design characteristics of the architecture are discussed below.

```
<context>
  <time class="time.TimeServer"
fact_file="time_facts.xml" />
  <location class="location.LocationServer"
fact_file="location_facts.xml" />
</context>
```

Figure 3. Context manager registration

```
<service>
  <message class="server.MessageService"
        action_file="message_actions.xml"
        />
  <file class="file.FileService"
        action_file="file_actions.xml" />
</service>
```

Figure 4. Service manager registration

```
<rule>
    <fact context="time" description="time is
      in range" method="isTimeInRange"
      parameters="15:00-16:30" />
    <fact context="location"
      description="user's location is at"
      method="isUserLocationAt"
      parameters="/ucc/prefab/prefab 1"
      />
    <action service="message"
      description="alert" method="alert"
      parameters=""Time to work!"" />
</rule>
```

Figure 5. Rule file example

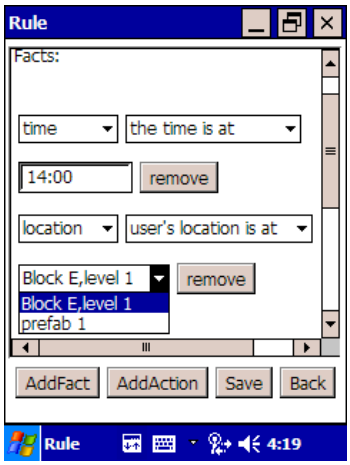

Figure 6. The DMS-CA user interface

2.1 Generality

The system is general in that it can be instantiated for an application context that deals with a particular building with certain types of sensors, and where particular services are offered. The definition of the context elements (such as location, sensor values, time, user schedule) and the services that can be invoked by a rule can be done by the system administrator using XML specifications, Examples of context and service registration are given in figures 3 and 4 respectively.

An event-driven architecture, where various contexts and services can be registered in a standard manner supports this generality. The actual rules defining which services a user requires for a given contextual environment can also be defined using XML (cf. figure 5) but are more easily done by the user on the mobile device using an

automatically generated interface (shown in figure 6). Thus the generic architecture can easily be deployed for a particular building and an individual user.

2.2 Efficiency

The DMS-CA makes each rule subscribe to the context it is logically related to, and the rule engine is designed so that only rules which can be fired by a particular context event are checked for that event. Time, being linear, is a special context and the DMS-CA has a built-in time task queue to efficiently check the time context rules. The DMS-CA then generates an efficient way of checking these rules in order to provide an effective means of context-sensitive delivery of data or executing other user prescribed actions.

2.3 User Transparency

In developing a smart environment, such as a smart building, care must be taken to ensure user transparency, i.e. that the particular rules of the system are available in a clear and explicit form to the user, and that a user can obtain a user-friendly explanation of actions taken by the system. The DMS-CA supports user transparency by allowing the user to define the actual rules that are used and do so in a user-friendly manner. The user can use the generated interface on the mobile device (such as a PDA) to define the set of context rules and services for its environment (cf. figure 6). In addition transparency is supported by the explanation and history options. These allow a user to obtain an explanation of a particular action (i.e. a user-friendly text describing the rule that caused the action) and also to see a history of what actions have been executed up to now.

3. Implementation

The generic architecture is based on Java events, generic context servers and XML-defined application-specific rules, and is designed to be used for a wide variety of smart environments. The DMS-CA is also a component within the general DMS [7, 2] architecture. Given that contextual information is changing constantly, one needs to avoid unnecessary checking of rules at every context change. To implement this, a technique of Event Driven Rule Checking is implemented. This method involves each rule defining a set of relevant contexts and subscribing to the relevant context changed events, so that the rule is only executed when these events are fired. Using this mechanism, rule tasks do not have to check their context constantly-they only need to do so when they are notified that a relevant context has been modified. Obviously this mechanism increases the efficiency of rule checking.

The time context is different from other contexts. Linear time means that the change events of the time context are predictable. A rule task subscribing to a changed event of a time context doesn't make sense, and therefore the time context is implemented differently. A special time task queue has been implemented to support all rule tasks that depend on the time context. Consider a rule such as: "If the time is 9:00am and I'm in the office, please download yesterday's building monitoring report to my PDA". This rule's condition involves both a time context and location context, but it's clear that it is only when the time is 9:00am that this condition has the chance to be true, otherwise it remains false. So the rule engine only has to check this rule at 9:00 am, which is once a day.

The time task queue mechanism is designed to minimize checking. When the rule task checks the time context for the first time, the time context will insert this rule task into the time task queue. The elements in the time task queue are sorted by the time specified in the condition of the rule task. The rule task which has the nearest upcoming time comes to the head of the queue. This sorting is done when a new rule task entry is inserted. The time task queue is implemented with a heap so that the queue achieves the most efficiency while doing work like inserting, removing, sorting.

The efficiency of the DMS-CA was evaluated by measuring its ability to process sensor events and process context rules using the time task queue. It was compared against a context-sensitive solution built using the software agent middleware, JADE [1]. The efficiency advantage of the DMS-CA is demonstrated in figure 7. Context rules were defined on the user's PDA, and the context-based rules were used to monitor room temperature. With JADE and other general purpose software agent based architectures a constant overhead is incurred due to the active threads associated with agents. With the DMS-CA the memory usage is considerably reduced as only timed events are examined.

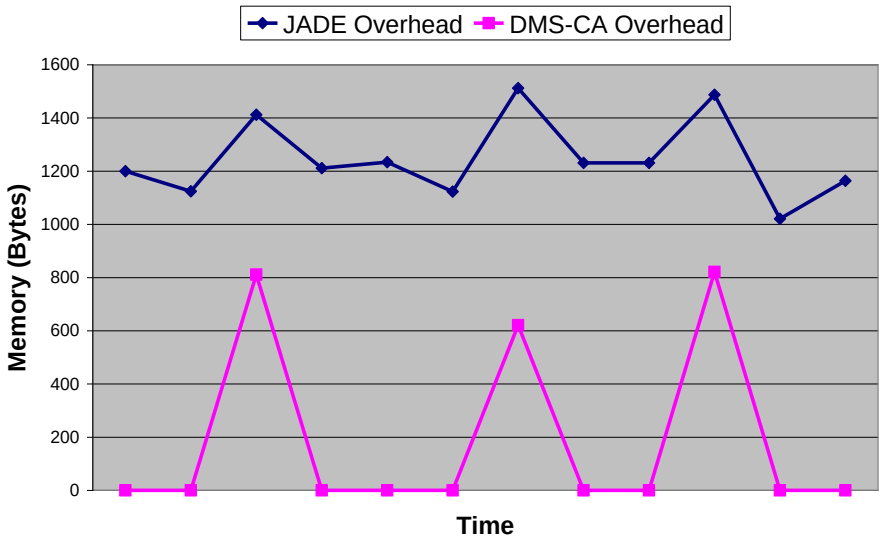

Figure 7. Comparison of JADE and DMS-CA

4. Conclusion

A smart building application can provide many benefits to users. The system presented has been developed as a completely generic system that can be tailored for particular applications. The novel aspects of the system include its combination of generality, efficiency and user transparency. The system has been developed in Java and deployed for an application with sensors based on the Tyndall mote and where a Dell Axiom PDA has been the target mobile device. An evaluation has demonstrated the efficiency of the system in comparison to an alternative agent-based solution.

5. References

[1] J. O'Donoghue and J. Herbert, "Data Management System: A Context Aware Architecture For Pervasive Patient Monitoring" in Proceedings of the 3rd International Conference on Smart Homes and Health Telematic (ICOST), pp. 159-166, 2005.
[2] J. O'Donoghue and J. Herbert "An Intelligent Data Management Reasoning Model within a Wireless Patient Sensor Network" Proceedings of Artificial Intelligence Techniques for Ambient Intelligence (AITAmI?6) in conjunction with ECAI 2006, Italy, August 2006
[3] T. O' Sullivan, J. O' Donoghue, J. Herbert and R. Studdert. "CAMMD: Context Aware Mobile Medical Devices". International Journal of Universal Computer Science, Special Issue on Pervasive Health Management: New Challenges for Health Informatics, 2006.
[4] N. Hristova and G. O'Hare, "Ad-me: A Context Sensitive Advertising System", Proceedings of 3rd International Conference on Information Integration and Web-based Applications & Services (II-WAS), Austrian Computer Society, Linz, Austria, Sept. 10th-12th. 2001.
[5] A. Hinze and A. Voisard, "Location- and Time-Based Information Delivery in Tourism" In Proceedings of SSTD 2003. pp.489-507
[6] G. Davenport and P. Pan, "Mobile, Context-sensitive Cinematic Narratives" MIT working paper. http://mf.media.mit.edu/pubs/journal/MobileCSNar.pdf 2006.
[7] A. Schmidt and C. Winterhalter " User Context Aware Delivery of E-Learning Material: Approach and Architecture" J. UCS 10(1): 28-36 2004.
[8] E. Pignotti, P. Edwards, and G.A. Grimnes "Context-Aware Personalised Service Delivery" European Conference on Artificial Intelligence, pages 1077-1078, 2004.
[9] E. Goh, D. Chieng, A. K. Mustapha, Y. C. Ngeow, and H. K. Low, "A Context-Aware Architecture for Smart Space Environment", Proceedings IEEE Multimedia and Ubiquitous Engineering, pp. 908-913, 2007.
[10] T. Hofer, W. Schwinger, M. Pichler, G. Leonhartsberger and J. Altmann, "Context-Awareness on Mobile Devices - the Hydrogen Approach", Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS), 2003.
[11] S.S. Yau and F. Karim "A Context-Sensitive Middleware for Dynamic Integration of Mobile Devices with Network Infrastructures" Journal of Parallel and Distributed Computing Volume 64, Issue 2, February 2004.
[12] F. Bellifemine, A. Poggi, and G. Rimassa, "JADE - A FIPA compliant agent framework" In Proceedings of PAAM, 1999.

Authors

Dr John Herbert is a Senior Lecturer in the Department of Computer Science, University College Cork, Ireland. His research interests include formal modeling and analysis of digital systems, real-time systems, quality properties of pervasive computing applications, medical applications of wireless sensors.

Dr John O'Donoghue is a Lecturer in the Department of Business Information Systems, University College Cork, Ireland. He received his PhD from the Department of Computer Science, University College Cork (2008). His research interests include pervasive data management, data and information quality, health informatics and medical based information systems.

Xiang Chen has a degree in Software Engineering from East China Normal University, China, and has completed the MSc in Software Development for Computer Networks in the Department of Computer Science, University College Cork, Ireland (2008), achieving 1st class honours.