Adaptivity support at the architectural level

Svein Hallsteinsen, Terje Løken, Tor Neple
SINTEF Telecom and informatics
N-7465 Trondheim
+47 73593000
{svein.hallsteinsen | terje.loken | tor.neple}@informatics.sintef.no

ABSTRACT

Mobility is fast becoming a key feature in software development, and mobility support brings forward a set of new requirements and challenges to both existing and new applications. At the core of these challenges is a changing and dynamic operating environment, as the users move from different terminals, locations and contexts.

To address these challenges, applications need to be increasingly adaptive, changing to fit the environment and the user's needs. This paper presents ongoing research aimed at introducing adaptivity at the software architecture level, based on ideas from component-based software engineering. We propose to use components with *adaptation interfaces* in addition to the traditional service interfaces. Intelligent agents can then be used to manipulate the architecture in order to achieve the desired functionality.

The approach means that existing systems built on static architectures can be endowed with increased adaptability through introducing components with adaptation interfaces, and agents capable of interacting with the architecture.

INTRODUCTION

Recent market analyses confirm the view that mobility is one of the key trends in ICT today [2], [3], [6]. This trend will bring forward new requirements to existing applications or systems and will also bring requirements for new types of systems. The overall goal being that the system masks from the users the fact that the user is mobile.

Mobility is about *moving around*. Mobility can be defined with respect to a users' movement in three spaces:

1. *The real geographic world.* This space consists of physical locations (space-time co-ordinates). Movement here is when users change their physical position.
2. *The global information space.* This space consists of the huge number of programs and files residing on computers throughout the world which can be accessed by remote users. When a user changes the information services she uses or chooses another application this corresponds to movement in this space.
3. *Connection technology.* This space consists of all the terminals and communications options open to the user. Moving in this space corresponds to the users selections of equipment.

In a mobile system the context of the user and thus the context of the system will change over time. Changing contexts provides a challenge to the systems architecture in two ways: i) defining mechanisms and patterns that support adaptation to the context of the system, bridging existing and future technologies, and ii) developing methodologies and model sets that facilitate the design of such systems. Technology is available today that enables systems to be distributed across networks, and allows users to access systems in a variety of ways, also from mobile settings. But the ways of access are foreseen in advance, and specialized functionality is produced to deliver just that access point.

The authors of this paper are currently taking part in the Norwegian research project LAMA. LAMA has set out to research into a wide variety of aspects within mobile computing, spanning from innovative applications to network and infrastructure technologies. The authors of this paper are working within the LAMA research area of system architectures and methodologies.

CHALLENGES OF MOBILITY

While mobile applications share the same difficulties and challenges found in stationary applications, the nature of their environment imposes new difficulties not found elsewhere. There are three fundamental requirements to mobile applications [5]: Wireless networking, Ability to change locations, and unencumbered portability.

Wireless networking is a much less stable communication environment than cable-based networking. While traffic intensity and bandwidth may vary on physical cables, the variations are much larger in wireless communication. Low bandwidths and high error rates are common, and disconnections are frequent but often brief. Although improvements will be made, these issues are not likely to disappear in the foreseeable future.

When bandwidth is a scarce resource, it is very important to utilise this bandwidth effectively. A mobile unit needs to adapt itself to varying bandwidths with as little user interaction as possible, and ideally without the user noticing it at all. The goal is to achieve the best possible quality of service (QoS) perceived by the user, and perhaps even more important, achieve a *predictable* quality of service from the user's viewpoint. This means that the quality of service should degrade as gracefully as possible when the quality of communication drops.

To make the mobile units less dependant on constant communication, one common solution is to move data and processes to the local client through explicit synchornization. This should rather be done automatically, where the mobile unit is able to decide what and when to update depending on the quality and availability of the network connection. Data sources and user services will have to be distributed and mobile, making it possible to mirror and move data and services to reflect user movement. However, this in turn leads to challenges related to consistency and availability, requiring flexible transaction mechanisms, automatic updates and intelligent prefetching.

While the users own data should be available independent of location and context, there is also a large volume of context-specific data that should be available to the mobile unit only during specific intervals in time and space. These include maps, special offers at the nearest supermarket, the location of the philosophy section in the library the user is in, and so on. These kinds of data need to automatically make themselves available to the mobile unit, and know when they are needed.

To achieve unencumbered portability, mobile units will have to be small. This places restrictions on the kinds of interfaces available on the unit, but a user will still want the same basic levels of functionality he or she enjoys in their office or their home. In addition, applications will be used both in mobile and stationary settings, with different equipment available. Context-aware applications will become important, as the mode of interaction will vary according to the context the user is in. As a simple example, voice communication is not always appropriate, but it is very appropriate if the user is using his hands for other tasks, such as driving.

As well as the challenges tied to the aforementioned requirements, the problem of security in a mobile application is affected by all the requirements. Communication should be shielded from unwanted listeners, and data both in the local unit and central data stores must be secure and available at the same time. Secure authentication is required at many levels, the mobile unit must not be useful for anyone other than its intended user, one must ensure that roaming data arrives at the intended server, a server must know that the mobile unit requesting data is who it claims to be, and so on. Making matters even more complicated is the fact that one cannot guarantee a predetermined level of security, and the quality of service should scale with different levels of security, rather than being an all-or-nothing proposition.

All of these issues have one thing in common: They demand an increase in the *adaptivity* of the applications used in a mobile setting.

Adaptivity

Adaptivity is the fundamental mechanism which will enable applications to function in the ever-changing world of mobility. Adaptivity can be introduced at all levels of an application, and may take many different forms, from adaptive protocols (packet size, rate of transmission) to adaptive user interfaces and adaptive qualities of service.

Common to all kinds of adaptivity mechanisms is the fact that they attempt to adjust some level of the application to account for changes in the operating environment. However, it may be difficult to analyze and reason about the adaptivity in the application *as a whole*, when adaptivity is introduced on an ad-hoc basis. Therefore, we believe that there is a fundamental need for software architectures that are adaptive in nature. This is closely connected to the ideas of reflective and adaptive middleware presented by Blair [4], where it is argued that mobile applications need to be aware of the implementation of middleware components, and need to be able to change the implementation to achieve the necessary adaptation. An adaptive architecture will allow reflection to be used effectively and consistently throughout the application.

APPROACH

Our background is in software engineering for evolutionary systems, i.e. systems that are designed for continuing change in order to cope with changing requirements and to be able to exploit advances in underpinning technologies.

In our opinion, the most promising approach to building systems capable of meeting such challenges is Component Based Software Engineering.

A system built according to this approach consists of a collection of software components collaborating through carefully designed interfaces and supported by a standardized component infrastructure providing the basic mechanisms for creating and destroying and communicating between components in a distributed hardware environment

This component-oriented architecture achieves its flexibility in various ways, for instance

- Separation of concerns, i.e. different aspects of the system isolated in different components localizes the effect of adaptation,
- Interfaces that are generic w.r.t. the anticipated changes facilitates adaptation by replacement,

- Components designed for adaptation through various forms of parameterization.

Basically we believe that a similar approach is applicable to the engineering of mobile systems.

As explained above, a key characteristic of mobile systems is the need to change in order to cope with varying communication and usage situations. However, unlike evolutionary systems where the adaptation usually takes place at compile or build time when producing enhanced or customized releases, the changes to mobile system must take place more dynamically while the system is actually being used, and preferably automatically and with a minimum of inconvenience for the user.

This calls for extensions to the architecture. Below we discuss some possibilities that we are exploring in LAMA.

Adaptation interfaces

One approach to coping with adaptation in evolutionary systems is known as Open Implementation [7]. The idea is to relax the black-box nature of components by providing them with a specific adaptation interface in addition to the normal service interface. The purpose of the adaptation interface is to offer the client some influence over the implementation choices of the component, without exposing implementation details. Many of these meta-interfaces already exist in some form, but require manual reconfigurations by a human system administrator.

Open implementation does support dynamic adaptation and appears like a suitable mechanism for coping with the kind of changes needed for mobile systems [4].

Adaptation agents

Just as software agents can be used to provide applications such as network management with intelligent automation of certain tasks, agents can also be used to endow an architecture with intelligent adaptivity. Andersson [1] outlines how to implement reactive software architectures using agents. Using components equipped with service adaptation interfaces, the agent is able to dynamically alter the architecture according to predefined rules.

Agents can interact with an architecture using triggers, effectors and sensors. A trigger is an event generated in the architecture, for example a loss of connection. An effector allows the agent to actively change the architecture, for example by cutting or creating connections between components. Sensors allow the agents to collect information from the architecture, which in turn can be used to make qualified decisions.

Collections of agents can cooperate or work independently to achieve the desired levels of adaptivity in the architecture. One possibility is to assign responsibility for different aspects of the system to different agents, with one agent being responsible for security, one for reliability of communication, and so on. An interesting starting point would be to construct an agent responsible for maintaining the quality of service in an application, by rerouting connections, moving data to and from the mobile unit, etc.

Adaptivity supporting middleware

Most modern information systems are built on a multi-tier architecture paradigm making use of some sort of middleware. In mobile systems it is essential that the middleware is adaptive. Through the LAMA project focus will be set on identifying the different requirements for adaptivity in middleware. In [8] issues regarding mobility and CORBA are discussed. This work within the OMG will be closely monitored and partaken in through the LAMA project. The goal of gathering these requirements is to be able to choose or build middleware with the necessary support for adaptivity.

FURTHER WORK

Using software agents to increase the adaptivity of an architecture creates many interesting opportunities for further research and refinement. We are also interested in developing methodologies and model sets that facilitate the design of systems using the proposed architecture.

Another interesting possibility is to employ adaptivity agents with learning capabilities, enabling them to learn which adaptations are most successful under different conditions.

REFERENCES

1. Andersson, J and Lennartsson, B. Reactive Architectures with Architectural Agents *Proceedings of 32nd Hawaiian International Conference on System Sciences, 1999*
2. The Business Implications of Mobile Computing and Sybase's Complete Solution; 1997.
   http://www.sybase.com/products/system11/workplace/mobile.html
3. Berst, Jesse; Why the world is going mobile; ZDNet; 12.12.1998.
   http://www.zdnet.com/anchordesk/story/story_2909.html
4. Blair, G.S. The Role of Open Implementation and Reflection in Supporting Mobile Applications. *Proceedings of DEXA'98*, W07: Mobility in Databases and Distributed Systems, http://www.ct.monash.edu.au/DPMC/mdds/mdds98/dexa98-toc.html
5. Forman, G.H. The Challenges of Mobile Computing *UW CSE Tech Report #93-11-03.* Available from ft.cs.washington.edu
6. IBM Application Framework for e-business, Mobile enablement; 1998.
   http://www.software.ibm.com/ebusiness/mobile.html
7. Kiczales, G. Beyond the Black Box: Open Implementation. *IEEE Software, January 1996*
8. Raatikainen, K. (ed.). White Paper on Wireless Access and Terminal Mobility in CORBA. *OMG document telecom/98-11-09*