Global adaptation framework for quality of experience of mobile services

Reference:
Sun Hong, De Florio Vincenzo, Gui Ning, Blondia Christian.- *Global adaptation framework for quality of experience of mobile services*
Proceedings of the 2007 IEEE Three-Rivers Workshop on Soft Computing in Industrial Applications - s.l., 2007
Handle: http://hdl.handle.net/10067/662150151162165141

Global Adaptation Framework for Quality of Experience of Mobile Services

Hong Sun, Vincenzo De Florio, Ning Gui and Chris Blondia
University of Antwerp
Department of Mathematics and Computer Science
Performance Analysis of Telecommunication Systems group
Middelheimlaan 1, 2020 Antwerp, Belgium, and
Interdisciplinary institute for BroadBand Technology
Gaston Crommenlaan 8, 9050 Ghent-Ledeberg, Belgium
{hong.sun, vincenzo.deflorio, ning.gui, chris.blondia}@ua.ac.be

Abstract

Nowadays, technologies are providing the mobile terminals with the ability to access to information and multimedia services from almost anywhere at anytime. However, the challenge is that in mobile applications, the surrounding environments are continuously changing, e.g. services are entering and leaving, quality of communication channels are changing, battery is decreasing, etc. In order to obtain high quality of experience, it is important to detect these changes timely and make adaptations accordingly.

Adaptation schemes are developed focusing on different specific domains, but there might be conflicts between each other thus jeopardizing the overall optimization. A global adaptation framework is urgently required to achieve global optimization, trading off among the concurrent schemes. In this paper, we are proposing such a global adaptation framework to efficiently detect the changes of system status and carry on adaptations in a global way, provide both coarse grain adaptations and fine grain ones, make best effort to eliminate possible conflicts between different adaptation schemes, deliver the fuzzy control in adaptation. Through this scheme the user's quality of experience can be greatly enhanced.

1. Introduction

With the development of technologies, functions of the mobile terminals are becoming more and more powerful; the accesses to the pervasive services are available in many areas. Users with powerful mobile terminals are granted with the ability to access to information and multimedia services from almost anywhere at anytime.

However, there are still some restrictions limiting the users' quality of experience: examples include a battery life which is not long enough for supporting the burst of power consumption, an environment that is changing when the user is roaming thus degrading the communication quality, etc. In order to increase the user's quality of experience, the mobile terminal should intelligently detect the changes of environment (battery capacity, communication channel quality, etc.), and take proper adaptations promptly.

Adaptations strategies are already developed in different domains: in the power consumption domain, reducing the power could obtain a longer battery life, while in the communication domain, increasing the power may guarantee a better communication quality. Adaptations in different domains may conflict with each other, thus jeopardizing the overall optimization. A global adaptation framework could help to coordinate the different adaptation schemes and alleviate the possible conflicts between the different adaptation schemes.

In order to achieve the global optimization, the different optimization schemes should be designed as distributed applications and under the coordination of one special management layer. Middleware is developed in the past decades which mediate the interactions between the application and operating system, hide the details of the underlying layer, and aid the development of distributed application for the new computing environment [1] [2]. Traditional middleware

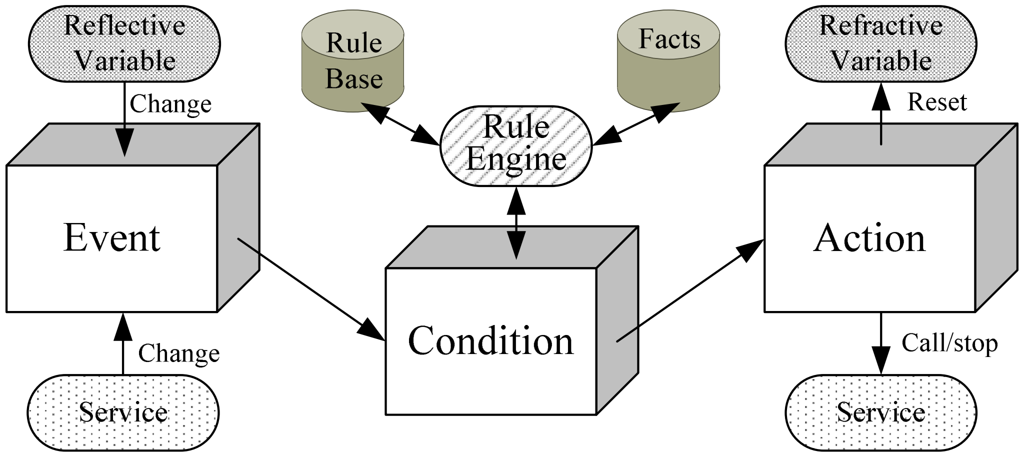

Figure 1. Architecture of Global Adaptation Framework

lacks the support for dynamicity, which is called for by the requirements of the current computational environments. Adaptive and reflective middleware is developed providing the system with the ability to reason about and act upon itself. However, the ability of the dynamic reconfiguration is still not satisfactory enough. Grace, et al. developed a sensor middleware for flood monitoring in 2006, which is called Gridkit. They claimed that among the existing sensor middleware at that time, “none of these pieces of work considers dynamic reconfiguration and customisability to the same degree as Gridkit” [3]. But even their sensor middleware only switched between two preset modes. In the mobile application, the contexts of environments are diverse and these contexts are also changing frequently. The adaptation framework should take appropriate adaptations promptly, and this is more challenging than for the flood forecast sensor network.

In this paper, we are proposing a global adaptation framework to fulfill the above mentioned challenges. We will use the service oriented architecture and reflective and refractive variables [4, 12] to realize the reflection and adaptation, in both coarse grained and fine grained aspects. Rule engine and facts base will be developed to implement adaptation rules.

The following of this paper is developed as follows: section 2 will generally introduce the structure of the proposed global adaptation framework; its three main components, i.e., the event, condition, and action module, will be introduced in section 3, 4, and 5 respectively. Conclusions shall be found in section 6.

2. Global adaptation framework

The architecture of the global adaptation framework is shown in Figure 1. In order to make the correct adaptation in the mobile application, there is a stringent requirement for the system to be aware of the environmental changes. The idea of context awareness is developed so as to deal with linking changes in the environment with computer systems, and sense the changes of the physical environment. Context is defined as "any information that can be used to characterise the situation of entities."[5].

The proposed framework is thus constructed as an event-condition-action model [6] [7]. The system is context aware, and adaptations are taken when the context changes. The event module detects the changes of the surrounding environment. The detected events will trigger the reasoning of the rule engine; depending on the conditions, specified adaptation rules will be fired and the adaptations will be held in the adaptation module. The following sections will introduce these three modules in details.

3. Event Module

The event module is designed to monitor the environment around the mobile terminal, availability of applications and the system status of the mobile terminal. The information that needs monitoring includes changes on application availabilities and values of indicative parameters, e.g. the availability of WiFi service, the amount of battery power still available. In the adaptation framework in this paper, we use services and reflective variables to separate different kinds of information.

Service oriented architecture is used as an approach to dynamically detect and employ services outside a term-

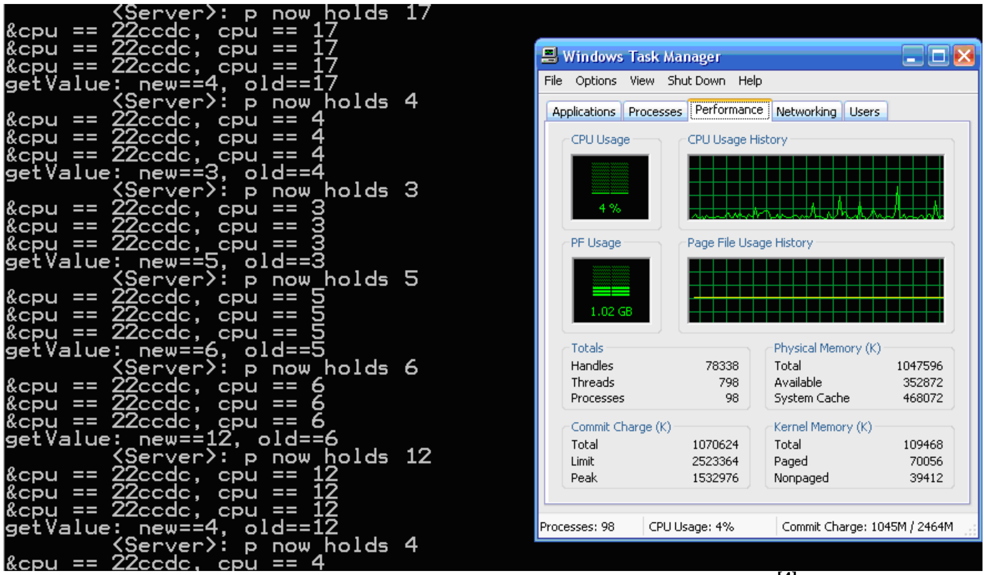

Figure 3. An Excerpt from the execution of the code[4]

-inal (external resources). We use the OSGI framework [8] constructing services as bundles, which can be remotely installed, started, stopped, updated and uninstalled without requiring a reboot. Changes of the services could be detected by the service listener. The code that detects the service change is as follows:

```
  public void
serviceChanged(ServiceEvent event) {
            switch (event.getType())
{
            case 1:
                  …
                  break;
            case 2:
            …
            break;}
```

Figure 2. Change detection in SOA

In Figure 2, *event* indicates a service which is being tracked, and the *event.getType()* function indicates which kind of changes has happened. A switch statement controls the corresponding operations to each change.

Although the service listener tracks the availabilities and changes of the services, this service awareness is only a coarse awareness of the environment; most of the times, the mobile terminals require not only the availability of services, but also the details of the services and the status of the mobile terminal and the surrounding environment. Some of the information maybe described in the meta-data of the service description, while some are represented by some of the key system state parameters, and we name those variables which reflect these statuses as reflective variables [12]. We have developed a translation tool that allows making use of reflection of the system status and external devices in a standard programming language such as C. The values of the reflective variables get asynchronously updated by service threads that interface with those devices. Figure 3 shows the execution of a code using reflective variables to track the value of the local CPU.

4. Condition Module

The condition module in this framework is implemented by the Jess rule engine [9]. Jess is a portable library which is used as the “brain” of a mobile agent. It has the capacity to "reason" using knowledge supplied in the form of declarative rules. Jess is small, light, and one of the fastest rule engines available. Its powerful scripting language gives access to all of Java's APIs.

The condition module is organized by rule base (knowledge base), working memory (facts), and inference engine (rule engine). The rule base contains user defined declarative rules, the facts are instances of templates, which could also inherit from Java classes, and be stored in working memory (it could be treated as repository). The inference engine will match the facts against rules, fire rules and execute the associated actions.

Our conjecture is that conflicts between different adaptation strategies may be detected by running inference rule in Jess. In the Jess rule engine, when the rule is fired, corresponding actions will be taken, each action will be interpreted as operations on the context attributes or changes of services. By detecting whether there are contradictory actions on the same fact (target), the possible conflicts could be detected [10]. Jess could control the execution orders of the fired rules with priority setting which is named as salience. In [10], conflicts in a home environment are investigated when more than one user tries to access a same target. The conflict is resolved by assigning different priorities to different users. In our condition module, we will assign adaptive priorities to different adaptation rules according to the context information. For example, the power saving rules will have low priority when the battery remaining is high, but when the battery drops to low, the priority will be automatically adjusted and set to high.

In the adaptation of the mobile terminal, one significant feature of the mobile computation is the imprecision of contexts. The mobile terminal is weak in computation power in nature, and the surrounding environment of the mobile terminals is often frequently changing. It is difficult to get accurate and precise context information. Soft computing is required to exploit the given tolerance of imprecision, partial truth, and uncertainty in this situation.

Fuzzy logic is used in the condition module to control the indistinct transitions. FuzzyJess [11], which is part of the FuzzyJ Toolkit, is a powerful set of fuzzy logic tools providing the Jess rule engine with the ability to reason about the fuzzy logic. A fuzzy rule adapting to the lower power mode when the battery level is low could be written as follows:

```
(defrule battery-low-set-power-low
           (battery ?t&:(fuzzy-
match ?t "b_low"))
=>
  (assert(power(new
FuzzyValue ?*powerFVar* "p_low"))))
```

Figure 4. Fuzzy rule

In the code in Figure 4, the FuzzyJess Function fuzzy-match compares two FuzzyValues, *battery* and *b_low*, to see how well they match, if there is some degree of matching, then the `battery-low-set-power-low` rule will fire, and a power fact will be asserted with a *p_low* FuzzyValue to reset the power. This value could be scaled by the degree of matching the battery FuzzyValue, e.g., when the battery is slightly low, i.e. 40%, the power will be set to moderately low power mode, i.e. 80% of the full power; when the battery is extremely low, i.e. less than 10%, the power will be set to very low level thus realizing the fuzzy control.

5. Action Module

The action module in the frame work will execute the adaptations when certain adaptation rules are fired in the condition module. Adaptations in the action module are executed in two ways:

Firstly, the coarse grain adaptation. The adaptation would be realized by using or stopping certain services. As illustrated in Section 3, applications will be designed as OSGI bundles, and one of the advantages of designing application in the form of OSGI bundle is that it could be start/stop/update without rebooting the system.

Secondly, the fine grain adaptation. The adaptation would be realized by setting some of the system parameters, which are named as “refractive variables” [12]. A write request will trigger a request to update a variable, and we use the control of these variables to realize more precise adaptations.

6. Conclusion

This paper proposed a global adaptation framework to organize the adaptation in mobile terminals, which aims at improving the user’s quality of experience. The framework is organized as event-condition-action model, and is flexible to take adaptations acting upon the changes of environment context. The improvements brought by our proposed architecture are as follows:

- The proposed architecture provides an agile way to detect the change of context by the combination of service change detection and the reflective variables.
- The proposed architecture alleviates the conflicts between different adaptation strategies by detections of Jess inference rules.
- The proposed architecture copes with the imprecision and fluctuation of the information with the fuzzy logic which is provided by FuzzyJess.
- The proposed architecture provides fine grained adaptation by the combination of changing services and resetting the refractive variables.

We are developing the proposed global adaptation framework in the IBBT project “End to end Quality of Experience” [1] to construct the end-to-end co-optimization infrastructure for quality of experience mobile services. The global adaptation framework will help to enable the deliver of cost-effective interactive media service while also meeting the user’s quality of experience. We will use this adaptation framework to coordinate adaptations between different coding methods, crosslayer optimization [13], and applications of Network Intelligence Proxy server (NIProxy) [14], and alleviate possible conflicts between these techniques.

Further work will also explore the use of this adaptation framework on thin-client application [15]. We will focus on developing rule base libraries for the thin-client applications. Rule bases will be developed with different user’s preference and also taking considerations of the surrounding environment of thin-client servers. The system will choose the appropriate rule base when changing thin-client server, or choose the rule base manually by the preference of the user, thus bringing the most effective adaptation, thus bringing the most effective adaptation or deliver best user’s quality of experience.

6. References

[1] F. Kon, F. Costa, G. Blair and R. Campbell. The case for reflective middleware: building middleware that is flexible, reconfigurable, and yet simple to use. Communications of the ACM (CACM), Volume 45, Issue 6, 2002, 33-38.

[2] G. Coulson. What is Reflective Middleware, *IEEE Distributed Systems Online*, 2000.

[3] P. Grace, G. Coulson, G. Blair, B. Porter and D. Hughes. Dynamic Reconfiguration in Sensor Middleware, In Proceedings of the 1st International Workshop on Middleware for Sensor Networks (MidSens '06), co-located with Middleware 2006, Melbourne, Australia, November 2006.

[4] V. De Florio & C. Blondia. Reflective and Refractive Variables: A Model for Effective and Maintainable Adaptive-and-Dependable Software, in the Proceedings of the 33rd Euromicro Conference on Software Engineering and Advanced Applications (SEEA 2007), Luebeck, Germany, 2007.

[5] A. Dey. Understanding and Using Context. Personal and Ubiquitous Computing. Journal, 5(1):4–7, 2001.

[6] R. Etter, P. Costa and T. Broens. *A Rule-Based Approach Towards Context-Aware User Notification Services.* In: Proceedings of the IEEE International Conference on Pervasive Services 2006, 26-29 Jun 2006, Lyon, France. pp. 281-284.

[7] V. De Florio & C. Blondia. A System Structure for Adaptive Mobile Applications, In Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2005 (WOWMOM 2005), Giardini Naxos, Italy, 2005.

[8] Open Service Gateway initiative (OSGI), OSGI Service Platform Version 4, http://www.osgi.org, May, 2007

[9] E.J. Friedman-Hill. Jess in Action: Java Rule-Based Systems, Manning, 2003

[10] I. Park, D. Lee, S. Hyun. A dynamic context-conflict management scheme for group-aware ubiquitous computing environments, In Proceedings of the 29th Computer Software and Application Conference, (COMPSAC 2005), Edinburgh, UK, 2005.

[11] B. Orchard, FuzzyJ Toolkit and FuzzyJess, http://www.iit.nrc.ca/IR_public/fuzzy/fuzzyJToolkit2.html, May, 2007.

[12] V. De Florio, J. van Vaerenbergh and C. Blondia. “Adaptive feedback-loop Control System for Inter-reality Management”, in Proceedings of the 2007 IEEE Three-Rivers Workshop on Soft Computing in Industrial Applications, Passau, Germany, August 1-3, 2007.

[13] W. Eberle, B. Bougard, S. Pollin, F. Catthoor. From myth to methodology: cross-layer design for energy-efficient wireless communication. DAC 2005: 303-308

[14] M. Wijnants & W. Lamotte. The NIProxy: A Flexible Proxy Server Supporting Client Bandwidth Management and Multimedia Service Provision, In Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2007 (WOWMOM 2007), Helsinki, Finland, 2007.

[15] D. De Winter, P. Simoens, L. Deboosere, F. De Turck, J. Moreau, B. Dhoedt, P. Demeester. A hybrid thin-client protocol for multimedia streaming and interactive gaming applications, in Proceedings of NOSSDAV2006, the 16th International Workshop on Network and Operating Systems Support for Digital Audio and Video, Newport, Rhode Island, USA, 2006, pp. 86-92

[1] Official website of End-to-end Quality of Experience: https://projects.ibbt.be/qoe/