Summary: Composing Components and Services using a Planning-based Adaptation Middleware

SENTAYEHU SEIFU **RESONDRY ZAFIMIHARISOA** **HERVE FALCIANI STASSIA**

INTRODUCTION

Dynamic adaptation is particularly relevant in the domain of ubiquitous computing, which is subject to numerous unexpected changes of the execution context. For example, a mobile device is frequently roaming, and its applications have to be dynamically adapted to remain useful under new network conditions.

Planning based adaptation middleware refers to the capability of adapting an application to changing operating conditions

1. What is the previous solution?

The MADAM Middleware defines a domain and platform-independent flexible reference architecture that supports dynamic reconfiguration of both applications and the middleware itself. Furthermore it implements a flexible context monitoring, adaptation planning and dynamic reconfiguration framework that embodies much of the complex logic of dynamic adaptation.

We introduce an extension of the MADAM component-based planning framework that optimizes the overall utility of applications when such changes occur. The extended planning framework supports seamless configuration of component frameworks based on both local and remote components and services.

2. Why SOA is of interest for self-adaptive applications?

Because services are reusable and composable entities that can be dynamically exploited to improve the behavior of an application executed on a mobile device. Services in a SOA environment can be discovered and accessed without knowledge of the underlying platform implementation and hence can be exploited in the dynamic configuration of the applications. Adaptation in -MADAM is generally QoS-driven and In SOA, QoS properties are part of the Service Level Agreements (SLAs) that are negotiated between a service provider and its end-user consumers.

3. What is the motivation for this work, and its distillation into a research question?

Imagine a user carrying mobile devices move around in ubiquitous computing environments causing frequent and unexpected changes in the execution context of their applications.

MADAM planning is based on dynamic configuration of component frameworks. The extended planning framework supports seamless configuration of component frameworks based on both local and remote components and services. In particular, components and services can be plugged in interchangeably to provide functionalities defined by the component framework.

4. What is the purpose of an adaptation-planning framework?

It is to compute and evaluate the utility of alternative configurations in response to context changes, and to select a good one for the current context.

5. What is the new Proposal?

MUSIC is proposed which supports self-adaptation of ubiquitous applications to changes in the service provider landscape. It is the extension of MADAM with the concepts of SOA, and then we have a service planning with SOA. By exploiting knowledge about its composition and Quality of Service (QoS) metadata associated to the application components. The planning middleware evaluates discovered remote services as alternative configurations for the functionalities required by an application.

This part is focused in this part to present the additional strategic components that constitute MUSIC:

a. **The composite SOA** was integrated; it permits to the Adaptation Middleware to combine several SOA technologies using different implementations. The component **SLA Negotiation** is responsible of the negotiation of the requested property (with SLO). The

component **SLA monitoring** stores the SLA agreement and is responsible to remove them dynamically if the agreed service level is not satisfied. One important component of this composite is the **Service Discovery** which generates the plan of the remotes services.

b. **The component Plan Broker** federates the local Plan Repository with the components Service Discovery. The adaptation manager retrieves plans via its interface during the planning.

c. **The component Service Binding** is used by the component **Configurator** if the plan which contributes to new configuration to reconfigure the application refers to a remote service. The **Service proxy** created implements the disconnection detection algorithm and is also responsible for monitoring the dynamic QoS properties associated to the SLA contract agreed with the service.

This new design brings a very large scalability in the sense that there are more available remote services and components, as alternatives configurations. These maximize the utility of the application and then contribute to improve the QoS offer to the end user.

6. Validation

This design was validated with a case study, through its two scenarios gives a concrete example of how effective MUSIC could be. The different compositions and the corresponding requirement/fulfillment are materialized in terms of Providers with their associated contracts. At the middleware level the requested properties (cost, accuracy or battery) define the utility function and then, the alternative configurations, with their associated normalized utilities, explaining clearly how all can fit together.

7. Let us Improve MUSIC:

This extension has a very performed result but we can improve it more by processing with a distributed approach, meaning we can add an **externalized SOA composite** like that the management of several available services (more services level can be proposed) will not be limited by the device resources instead of delegate this task to a service proxy. A frequent dynamical service negotiation will not anymore a time critical factor because a large choice of service provider offering the same services is taking in account at the same time (parallelism) in the external component side.

A **rule** which give more **priority** to the local component, if it satisfied the requirement needed, must be added when the Adaptation Manager selects the plan which has the highest priority to contribute in the new configuration. It will guarantee the stability of the application because it avoids the impact of the intermittent availability of a remote service.

Threshold T < Utility Local < Utility remote services

➔ Priority for Local component.

8. Related Work

Related works were done on different or complementary approaches to MUSIC planning-based middleware:

a. Adaptive Service Grids, using semantic description of what is intended to be done rather than describing which services to use and how to select them in regards with parameters or SLA.

b. Menasce and Dubey, Closer to SLA a QoS broker represents an alternative where services providers must register, thus preventing its use in a mobile environment where the broker may not be available.

c. CARISMA rely on reflection and concentrates on conflict resolution thanks to an auction-like procedure.

d. Finally ReMMoC developed the ability to benefit from mobile client and ubiquitous services.

CONCLUSION

In conclusion the design of a QoS driven generic planning framework revealed itself intuitive and so, easy to apply in common business scenarios. The SLA protocol is of course the core part of this architecture. However limitations of such design were studied by us during this work such as the instability of the application increasing the consumption of resources, when the service is intermittent, and we obviously propose some solutions were proposed such as the addition of new rule which must give more priority to local component and an externalized SOA composite.