Coupling MDA and Parlay to increase reuse in telecommunication application development

Babak A. Farshchian Sune Jakobsson
Erik Berg
Telenor Research and Development
Otto Nielsensvei 12
NO-7004 Trondheim, Norway
{Babak.Farshchian, Sune.Jakobsson, Erik.Berg}@telenor.com

Abstract

MDA (Model-Driven Architecture) has been coined by OMG (Object Management Group) as the next step in application integration. Being based on standards already embraced by a large segment of the software engineering industry, MDA promises fully automatic model transformation. MDA enables application developers to use formalisms such as UML to specify their applications in a totally platform-independent way. Later transformations to platform-dependent software are automated. Parlay, a middleware specification developed for the telecommunication domain, is on the other hand promising network independent development and deployment of telecommunication services and applications. In this position paper we report on our experience from a Eurescom project where we try to couple MDA and Parlay in order to increase reuse in the telecommunication domain. Telecommunication service development is hampered by long development cycles and low level of reuse. We describe how MDA approach can be applied to telecommunication domain through the use of Parlay. We believe this approach has substantial potential for reducing development costs for many telecommunication operators. In addition, developed models and applications can be deployed on a wide variety of platforms without much change.

1. Introduction

Market situation in telecommunication is changing rapidly due to the convergence of Internet and telephony networks and the removal of monopoly situations in many countries [7]. This change poses great challenges and opportunities for telecommunication operators and service providers. One such challenge has been the increased competition in providing advanced value-added services to the customers [3]. The abandoning of monopolist market models has brought with it a pressure on incumbent carriers to make available their network resources to be used by third-party service and application providers [8]. This creates a specialization of the telecommunication application development business that again puts pressure on both network operators and third-party application providers to optimize their development processes.

Value-added services and applications in convergent networks increasingly contain a large software part, as opposed to traditional telecommunication services where software played a more modest role. Application developers resort to (often complex) software solutions for accessing and deploying network resources offered by network operators. Software is also used in order to create innovative applications that make use of a mixture of network resources (e.g. applications operating transparently on both Internet and telephony networks and utilizing corporate databases). Increased competition and the innovative nature of convergent networks make it necessary to deploy development processes that allow for rapid application development so that developers can experiment with different application types without binding too much resources. Issues that have been of central importance for the software engineering community, such as reuse and object-oriented software development, is becoming more and more important also in the telecommunication domain.

One of the main software engineering issues facing telecommunication application developers is reuse. Telecommunication networks have traditionally been proprietary, developed as a result of strategic alliances between equipment producers and incumbent network operators. Similar applications have had to be developed multiple times on each proprietary platform. The situation is not to be changed in the near future due to large invest-

ments in proprietary network technologies. Nevertheless, the increasing number and variation of offered services and applications make the situation critical. There is an urgent need to enable application developers to reuse their applications on different proprietary platforms.

Important initiatives are taken in order to address this issue. In this paper we will look at two of these initiatives, i.e. Parlay and MDA (Model-Driven Architecture). Parlay is a middleware specification developed specifically for the telecommunication domain. Parlay enables network operators to open up their networks and make available their network resources in a systematic and standard manner [8]. Parlay enables reuse at the component level. Although Parlay allows service providers to reuse their services on multiple networks, there is little support for reuse at the application modeling level. Another relevant initiative, started by the software engineering community, is OMG's MDA [12]. MDA addresses reuse at the business modeling level. MDA aims at increasing automation in software development by deploying automatic model transformation techniques. The goal is to specify applications in a business modeling language such as UML, and to have some tool automatically create the software.

Although both Parlay and MDA are in their early days, they have gained large industry support. Supporting tools are entering the market, and an increasing number of companies are deploying solutions based these technologies. In order to evaluate MDA we are participating in a Eurescom project (see [4]) together with a number of telecommunication operators and tool vendors. Our goal in this project has been mainly to evaluate MDA as a possible enabler for improving application development activities undertaken by many telecommunication companies. This paper describes some of our experiences so far, in particular those related to reuse at application modeling level.

The paper is organized as follows. First we give a short overview of Parlay and MDA. We then describe how MDA and Parlay can be used in combination in order to improve application development activities. A discussion of the implications on reuse is then provided before we conclude the paper.

2 Overview of Parlay

Parlay [8, 13] is a platform-independent, object-oriented middleware specification developed specifically for the telecommunication domain. Parlay is being developed by the non-profit Parlay group as an open specification (see www.parlay.org). Figure 1 shows an overview of how Parlay works.

The conventional approach to service provision in telecommunication networks allows only the network operator to develop applications (e.g. voice messaging, follow-

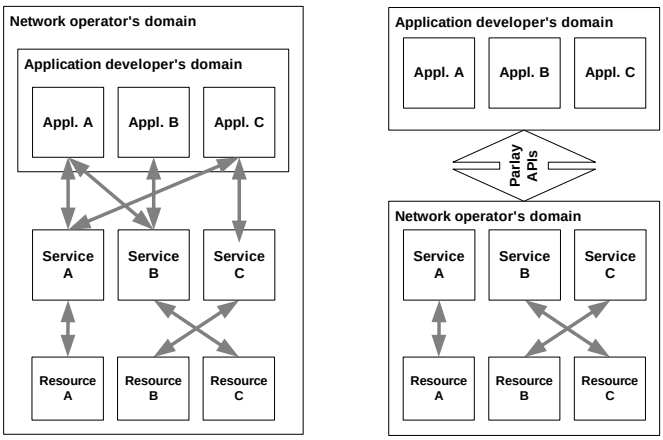

Figure 1. Parlay enables third-party application developers make use of network resources.

me functions) based on its own services. These services and applications are normally developed specifically for the network operator's proprietary network, and are not reusable in other contexts (Figure 1.A). Parlay defines a standard interface towards the network operator's environment (Figure 1.B). This environment includes resources such as fixed and mobile networks. Services that provide access to these resources are standardized through Parlay APIs. Application developers can call API methods on network operator's services and make use of the underlying resources in their own applications. This approach allows application developers to develop advanced applications utilizing a mixture of services and resources (e.g. fixed and mobile telephony, Internet, corporate databases). In addition, the approach increases reuse by making application specification independent of underlying network.

Parlay APIs' architecture is shown in Figure 2. The Framework Interface is an authentication and service discovery component that is a standard part of any Parlay implementation. This interface provides standard methods for authentication of third-party external applications that wish to make use of network services. Framework Interface also provides standard methods for publishing and discovering existing services in a network.

Service interfaces provide standard access methods to common telecommunication services. Parlay specification contains a large number of such service definitions (e.g. call control, session control, messaging, account management) but allows for new services to be added. An (external) application has to be authenticated before it can look for desired

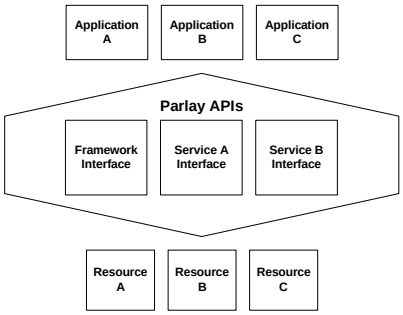

Figure 2. Parlay APIs' overall architecture.

services. Once contact with a service is established, the application can make use of the service. Parlay also specifies a callback mechanism that allows services to call methods on the applications (e.g. to notify an application if a call is made to a specific number).

Parlay enables reuse beyond what is available in current telecommunication networks. Service definitions can be reused. The same service definition can provide access to a variety of resource types. For instance, a call control service can be used to connect users using mobile, fixed, or Internet phones, without the service interface being changed for each type of phone. In addition, Parlay opens for reuse at the application level. Applications need to deal only with Parlay APIs. Application developers can specify their applications without taking into consideration what kind of network they will be deployed on.

Although Parlay enables reuse of components (i.e. services), it does not support higher-level reuse, such as reuse at the application model level or reuse of component compositions. What happens in the application level, e.g. what kind of business models are developed and reused, is outside Parlay's scope. This is where the MDA approach promises to be of benefit.

3 Overview of MDA

MDA (Model-Driven Architecture) is OMG's vision of enterprise application integration [2, 10]. MDA is based on model transformation principles, some known from earlier research and development within the CASE (Computer-Aided Software Engineering) community. MDA is a promising approach mainly because it is based on already-embraced industry standards such as UML (Unified Modeling Language) and XML (eXtensible Markup Language), which provide an organizational and technological springboard for the approach. In addition, the deployed standards already offer a level of formalism that makes it feasible to perform model transformations with a reasonable level of automation.

Figure 3 shows an overview of the MDA approach. At the heart of the approach is the Meta Object Facility (MOF) [9]. MOF is a meta metamodel that is used to define all the metamodels (i.e. modeling languages) in an MDA architecture. MOF is used to define UML, which is a business metamodel. MOF is often mapped into an OMG-defind XML standard called XMI (XML Metadata Interchange). XMI is used for exchanging models among tools, e.g. for transformation purposes.

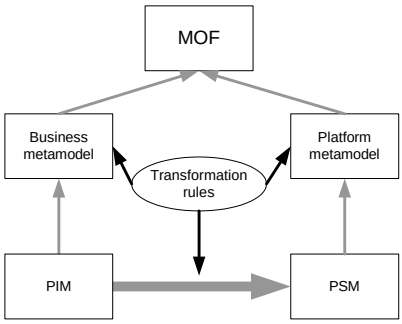

Figure 3. An overview of the MDA approach.

Any number of metamodels can be defined using MOF. MDA approach is based on developing separate metamodels for business domains (e.g. telecommunication) and technological platforms (e.g. Parlay). A set of transformation rules formally define how models defined using one metamodel can be transformed to models defined using another metamodel.

An application developer can use a business metamodel to model business applications. An application model is in this way developed to be totally independent of technological platforms the application will be deployed on. Such an application model is called a Platform-Independent Model (PIM). Platform-Specific Models (PSMs) are on the other hand defined specifically for the target platform. PSMs are ideally not developed by people but are generated automatically using proper transformation rules.

MDA is an enabler of reuse at application model and metamodel levels. Reuse at the model level is mainly due to platform-independent development of application models (i.e. PIMs), but also because of the possibility to reuse model segments and component compositions. The same PIM can be used to generate several different PSMs by simply defining new transformation rules. At the metamodel level, one metamodel needs to be developed for each business domain and each technological platform. Transformation rules themselves are of course reusable across metamodels.

Although MDA is notation-independent, OMG's recommendation and a large part of the development within the industry are based on UML. UML is particularly suited for the

MDA approach because of its extension mechanisms [1]. Business and platform metamodels can be defined as UML profiles (for an example of UML profiles see [6]). In addition, having a formal meta metamodel ensures consistency and interoperability.

4 Using MDA and Parlay to develop applications

Using the MDA approach in combination with Parlay is one of our main research goals. Such a combination has the potential to increase the level of reuse significantly. PSMs for each proprietary platform need to be developed only once, by the network operator or other software vendors. Third-party application developers will only need to develop PIMs for their applications and reuse these PIMs on multiple platforms, possibly belonging to multiple network operators. The overall process is shown in Figure 4. To the left of this figure we see the application developer who develops the PIM for the desired application. Transformation rules are used by the development tool to automatically create a PSM, which is an application directly written for Parlay. This application will make use of any network resource that offers a Parlay service interface.

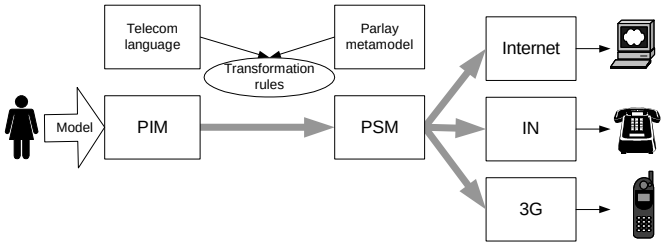

Figure 4. MDA-enabled development of telecommunication applications.

In order to enable this approach we need to start by defining a business metamodel (modeling language) for the telecommunication domain. This metamodel will be MOF-compliant and will focus on the specific needs of the telecommunication industry. Some of these needs are identified to be [5]:

- Telephony networks: UML needs to be extended with concepts that constitute the domain of telephony networks and related services. Examples of such concepts are call, conference, voice message, and telephone number. These concepts are independent of any underlying technology and should be specified in application PIMs.
- Telecommunication service access and subscription: UML needs to be extended with concepts to allow application developers to model how customers, customer profiles, services, access to services, subscription etc. will be managed for the purpose of billing and resource allocation. Access and subscription management is also mainly independent of the underlying technologies (except where there are differences in prices based on the used technology) and should be modeled in PIMs.
- Telecommunication network management: Assuring that a telecommunication network is functioning properly is crucial for the customers. A number of parameters, such as fault management policies, performance management, and security management can be modeled independently from the underlying platform.
- Quality of service: Quality of service is in many cases a part of the application domain, independently of what platform the application is running on. Some applications, e.g. emergency numbers, will require very high availability, while other applications e.g. video conferencing will require high performance. These parameters should typically be defined in a PIM.

Developing a metamodel for the telecommunication domain is a great challenge. Such a metamodel should have enough expressive power in order to satisfy both conventional telecommunication needs and the needs of the increasing number of services and applications that are based on convergent networks. Such a metamodel can be developed as a completely new modeling language, or it could be a specialization of UML. In both cases compliance to MOF is crucial in order to allow for interoperability with future MDA-enabled tools.

A platform metamodel, and corresponding transformation rules, will take as a starting point the Parlay API specifications. These specifications are already documented in UML, and it is expected that the influence of UML on Parlay specifications will increase as UML 2.0 is accepted by the industry. Compliance with UML will make it easier to develop transformation rules that can automate the transformation task to a satisfactory degree. In fact, the formal specification of UML 2.0, with the increased centrality of MOF, also makes it feasible to talk about full-scale roundtrip engineering.

In applying the scenario described above, development tools and environments will play a crucial role [11]. MDA-enabled tools are entering the market. This new generation of tools will have a meta-CASE flavor in that they will enable flexibility both in front-end and back-end in order to allow different business and platform metamodels to be used in combination. Defining architectures for such tools is an important part of our work.

5 Enabling reuse

To sum up, our approach will increase reuse in the following specific points:

- Reuse of application models: The business metamodel and the resulting application models will be totally independent of the underlying service infrastructures, and will focus solely on the business area to be supported by the telecommunication application. This is in accordance with the MDA vision. This means that when new services with better quality make their way to the market (e.g. 2.5G and 3G services) the application models can be reused without much change. In fact, we envisage UML packages that model basic telecommunication applications (e.g. number conversion, personal answering machines, voice mail) to be developed by third-party vendors.

- Reuse of application and architecture patterns: The approach will offer the possibility to reuse specific combinations/patterns of service components. Such patterns will offer a significant advantage for application developers who want to customize their applications to the needs of different customer groups.

- Reuse of service components: Services defined in form of Parlay service interfaces will be reusable in case of changes in the underlying network technologies.

- Metamodels and transformation rules: Metamodels and transformation rules developed during our research are generic and can be reused. In fact, they are planned to be incorporated into MDA-enabled tools developed by our partners. Knowledge transfer activities, in particular towards standards bodies (e.g. OMG) is also a high priority activity within our research.

6 Conclusions and future work

We have described our preliminary results from a European Eurescom project where we aim at coupling Parlay and the MDA approach to increase reuse in telecommunication application development. The approach is based on developing a MOF-compliant modeling language for the telecom domain that addresses the specific needs of this domain. Models developed using this language will be automatically transformed into Parlay applications using a set of transformation rules. The approach is ideal for third-party application developers who want to build applications on top of the network infrastructures offered by large network operators.

The work reported here will be developed further in a European IST project starting in 2002. Our future work in the context of this upcoming project is to develop the different components of the approach. Our particular focus will be on developing a business metamodel for the telecommunication domain, an architecture for MDA-enabled tools, prototypes of MDA-enabled tools, and methodologies for MDA-enabled application development within telecommunication. We also plan to run a number of experiments involving real world cases. These experiments will allow us to measure the gained amount of reuse, in particular with respect to component reuse, pattern reuse and application model reuse, which are the main expected advantages of using the combined MDA-Parlay approach.

7 Acknowledgements

The work reported here is a result of the Eurescom P1149 project. We thank all project members for cooperation and feedback on the ideas presented here.

References

[1] S. S. Alhir. Unified modeling language extension mechanisms. *Distributed Computing*, pages 29–32, Dec. 1998.

[2] J. Bézivin. From object composition to model transformation with the MDA. In *TOOLS' USA, Santa Barbara, CA USA*. IEEE, 2001.

[3] Y. De Serres and L. Hegarty. Value-added services in the converged network. *IEEE Communications*, 39(9):146–154, Sept. 2001.

[4] Eurescom P1149 project team. Impacts of changes in enterprise software construction for telecommunications, 2002. http://www.eurescom.de/mda4telecom.

[5] Eurescom P1149 project team. Model driven architecture - Adaptations and impacts for the telecom domain. Project deliverable, Eurescom, Heidelberg, Apr. 2002.

[6] M. Fontoura, W. Pree, and B. Rumpe. *The UML profile for framework architectures*. Addison–Wesley, Boston, 2002.

[7] R. M. Frieden. *Managing Internet-driven change in international telecommunications*. Artech House, 2001.

[8] A.-J. Moerdijk and L. Klostermann. Opening the networks with Parlay/OSA APIs: Standards and aspects behind the APIs. Draft of submitted article, 2002.

[9] Object Management Group. Meta Object Facility (MOF) v1.3.1. Specification, OMG, 2002.

[10] OMG Architecture Board MDA Drafting Team. Model driven architecture – A technical perspective. Technical report, OMG, 2001.

[11] J. Siegel and OMG Staff Strategy Group. Developing in OMG's model-driven architecture. Technical report, OMG, 2001.

[12] R. Soley and OMG Staff Strategy Group. Model driven architecture. White paper, OMG, 2000.

[13] The Parlay Group. Parlay APIs 2.1. Technical white paper, 2001.