Relating MDA and inter-enterprise collaboration management

Lea Kutvonen
Department of Computer Science, University of Helsinki
Lea.Kutvonen@cs.Helsinki.FI

Abstract

The goal of MDA (Model Driven Architecture) approach is to provide tool chains that support generation of application implementations, and interoperability of applications by ensuring that communication models can be shared by different components of distributed applications.

This paper discusses the relationship of MDA tools and components with the inter-enterprise collaboration management that has become crucial for the success of enterprises. The MDA components and tools are seen as localized, intra-enterprise elements, with structural requirements on shared abstract computing platform. That platform is expected to enable inter-enterprise business processes to be run, using the MDA provided components as participants. Essentially, the MDA tools are visioned as factories taking service descriptions and generating implementations, metainformation for local management services, and metainformation used for inter-enterprise collaboration establishment and management.

This relationship between MDA and inter-enterprise collaboration middleware induces needs for shared model and pattern repositories, and ontologies supporting queries from them. Furthermore, this relationship between MDA and process-oriented systems reserve MDA techniques on the (ODP) engineering level solutions, while (ODP) enterprise level descriptions are used as metainformation for collaboration middleware.

1 Introduction

The OMG MDA (Model Driven Architecture) [1, 2, 6] aims for tools and solutions that rise expressiveness of programming tools and provide interoperability of software components across platforms. The MDA approach uses a unified system model by taking the full application network and capturing it into a single (or composed) model, PIM (platform independent model). This model is then transformed (stepwise using several refinements and modification) into an (partial) implementation. Parts of the unifying model may be transformed using different set of transformation rules, giving a solution for a heterogeneous platform.

Looking at the emerging ICT support for inter-enterprise collaboration, the first necessary step is to the development of enterprise systems, and intra-enterprise business processes within them. For this work, MDA brings a welcome and necessary improvement. The three modeling layers – CIM, PIM and PSM – allow process-aware software components to be developed and interoperate because they are developed using the shared CIM model that represents enterprise operational needs.

However, the second step in enterprise system evolution is the adjustment to various business networks. New generation ERP systems, distributed or collaborative workflow systems, and inter-enterprise business process management systems require modeling of a "global" collaboration model within which partners have specific roles to be fulfilled by their ICT system services.

The inter-enterprise arena is not directly addressed by MDA. Still, MDA components (component used to refer to produced software components, MDA tools, transformation rules etc like in [4]) bring a significant element to the overall collaboration architecture.

The business networks can be established and managed in various ways, namely by integration, unification via shared model, or by federation. Integrated solutions are what we have seen in EAI and B2Bi solutions [13, 13]. Unified solutions trust on shared metalevel model for coordination and interoperability, like in MDA. In an inter-enterprise setting, MDA tools can be used directly, but only if the network of enterprises and their collaborative business processes can be designed together and participants are willing to replace their internal process components with new ones or are able to map new processes on top of existing services. Federated solutions require separate facilities to exist to provide an environment, a breeding environment [5], to find appropriate process models, negotiate of their use, and agree on participation on the established network together with terms and conditions of the operation.

Even in the case of federated solutions, and cases where the global business process is not used for generating executable elements, but for monitoring conformity, the actual service components need to be created with some tools. Here, MDA tools are very applicable, as the metainformation required by both facilities are of the same type.

Relevant points of design include the platform models assumed. In the following, the shared abstract computing platform for inter-enterprise business process management is briefly commented, and its effects on the structure of MDA components is discussed. Special attention needs to be placed for communication, and agreement on communication contents and context; the ODP viewpoints [8] provide a method for describing what platform elements and contractual elements need to be involved.

For the inter-Enterprise processes, choreographs between independent services are relevant. Therefore, the question arises on how MDA supports production of service implementations taking both the platform requirements and the service description (signature, behavior, NFA features) into consideration.

2 Idea(l) of shared abstract computing platform

The overall architecture model discussed here is the one used in web-Pilarcos project [12, 11, 9, 10]. In the model, a federation contract is formed to define the collaboration processes and roles between enterprise services. The B2B middleware carries responsibilities of running the partner discovery, contract management and behavior monitoring protocols. The service components are independent from each other and only required to provide the service denoted in terms of external behavior and information exchange. The autonomy of service providers is emphasized; the internal implementation or deployment aspects are strongly encapsulated.

In environments where enterprise applications become members of dynamically established inter-enterprise business networks, the following metainformation services are needed:

- identification of the intended network structure, involving the topology of the network for responsibility distribution and collaborative business process models;
- discovery of potential partners for the roles in the network;
- static verification of interoperability between communicating partners; and
- contract management (establishment, monitoring, exception management, termination).

The key element in the infrastructure is contract and contract management facilities. The business network contract (federation contract) captures the business process models involved and maps the roles presented in such a way that each participant has one and only one combined role in the network. The roles are populated using discovery service for suggestions, and by assuring the selected service offers present an interoperable network.

The essential part of the role requirement is that of provided set of services and required set of services from peers. Implementation requirements of the service may call for requirements on service from the local platform; all "side effects" of processing towards peers should be visible in the service. Some integration requirements may however be present: requirements for binding object support and requirement for the use of integrated repositories need to be set as specific service properties.

Figure 1 illustrates how these services can be seen as potentially external infrastructure services between enterprises. The requirement for each enterprise is to support interfaces for corresponding metainformation exchange protocols.

The model repositories (type repository and business process model repository) are to support static verification steps during the network population phase and during any repopulation events later in the network lifetime [17, 14]. Therefore, the MDA components need to be present in the operational infrastructure services. As the contract is phrased in platform independent terms, all participants need to be able to reflect their own solutions relationship to the abstraction. The repositories need to provide an open, incrementable set of transformation both horizontally and vertically. The existence of horizontal transformations (PIM-PIM, or PSM-PSM transformations) a) requires an underlying (implicit or explicitly stored) unifying model to exist and b) indicates an interoperability relationship to exist. The existence of vertical transformations a) support traversal of the relationship tree for analysis purposes and b) support code generation and dissemination of best practices knowledge.

The MDA transformation rules and transformation filters need to be stored into service, information representation, process model and NFA definition ontologies for runtime use. Verification of relationships is resource consuming task, and thus needs to be performed separately.

The network of relationships is built by a set of designers, filter programmers, ontology creators etc. New kind of infrastructure requires an enhanced set of new "professions" as described for example in [7]. In addition, standardization efforts providing standard collaborative processes (like RosettaNet PIPs [3] or proprietary supply chains processes) gain from a shared publication method online and thus easier adoption cycle.

For business network establishment, each enterprise provide metainformation via traders about the use of those ser-

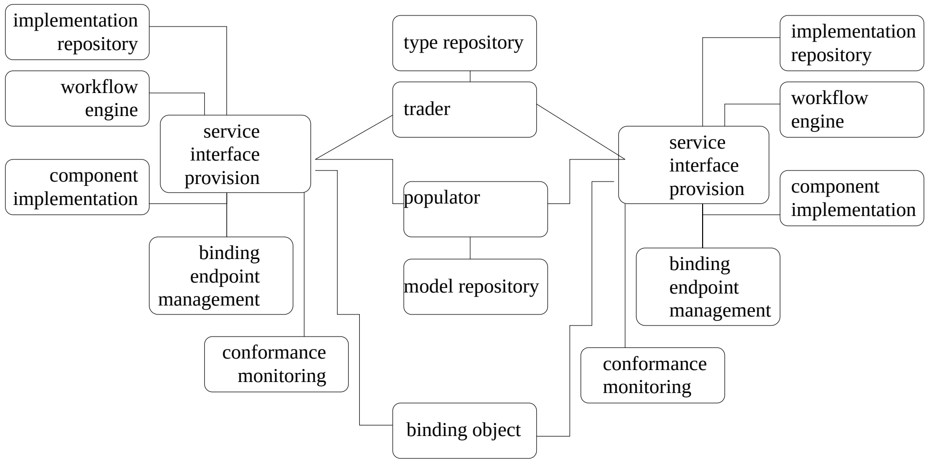

Figure 1. Architecture

vices they provide. Traders are supported by type repositories for resolution on whether two interfaces are alike, replaceable by each other, or not compatible. The populator fills in a business network with interoperable services.

For the runtime communication, the essential element in the architecture is that of distributed, open binding object. The object is constructed according to a binding contract, which declares the selected distribution transparencies, transaction choreographs, QoS agreements, and endpoint characteristics.

For the runtime verification of model conformant business process enactment, monitoring services are needed. Sensors can become standard elements of the alternative binding architectures. For the development of pervasive monitoring services, the language concepts for the monitored phenomenon need to be agreed on. This means ontologies of various aspects, like NFA features, dependent on each business process application domain.

3 Deriving requirements on PIMs from the business network environment

Within the architecture, three modeling points are of specific interest: the business network models, the external behavior models of service interfaces, and the model for the service realization within the enterprise. Here, the term realization is selected in favor of implementation, as the realization will often span a group of applications, data repositories etc.

The MDA processes to be used here would produce service realizations, starting from a set of models and producing appropriate implementation code (frameworks), and metainformation to be published in the B2B middleware repositories. The code should not include binding management, interoperability tests with peers, partner selection logic, or other elements provided by the B2B middleware. Instead, only the application logic should be present and conform to the external behavior model of its service interface type. Implementation must be parameterizable by NFA alternatives and other contract values.

The MDA road map from OMG describes MDA process with three model layers, CIM, PIM and PSM, roughly related to ODP viewpoints. CIM (computation independent model) relates to the enterprise viewpoint, PIM (platform independent model) to computational viewpoint, and PSM (platform specific model) to engineering model. The process is started from top, generating PIM models from the CIM models, with the advise of some transformation rules. Likewise, more detailed patterns advise the generation of PSM models from PIM models.

In the inter-organizational setting, the CIM model of focus describes the enterprise service internal logic that is externally visible through the provided service interface. Thus, in the MDA tool, a new CIM model needs to be verified against a published external service type. The set of enterprise business processes is more or less consistent and preplanned for efficient use of computing solutions. Modeling the processes and analyzing the processes as a set (BPA, BPR, etc) is an important goal in itself, especially combined with the view of inter-enterprise processes.

This CIM model can be further refined to PIMs. At the PIM level, also other models should appear in the enterprise model repository, namely those processes that support aspects of computing platform properties (security, trust management, QoS management, authorization, enterprise policies, service and binding factory management). These models should be prepared in such a way that aspects of behavior that can be negotiated within the inter-enterprise network can be configured either by selecting a suitable service component or by setting configuration attribute values.

For any enterprise service to be generated starting from a CIM model, a derivative PIM should be produced using selected CIM patterns. In addition, a set of PIMs that represent required computing platform properties should be joined with that business logic PIM for analysis. For code generation, the PIMs of computing platform properties should be dealt with as platform definition, giving the target concepts to be used by the implementation.

The binding elements should be provided as separate service elements. The production of these elements should go through the same kind of production process as the services within collaborative business processes.

Information or documents exchanged in the business processes are not described in all modeling techniques. However, modeling of information is an essential aspect that should indeed be modeled explicitly, and as a separate modeling issue. So, in addition to PIM models, there should be separate PII models (presentation independent information models) that can be mapped down to various representation formats. Transformations between representations of the same models could be placed as a responsibility of bindings.

When an enterprise service becomes deployed, it needs to be made available in the network. This is done by exporting appropriate service offers. For the proper establishment of dynamic business network contracts, the service offers need to capture metainformation that describes the service from several points of view, capturing the service description from ODP enterprise viewpoint and ODP computational viewpoint in respect of the actual service, and from ODP engineering viewpoint and ODP information viewpoint in respect to bindings.

4 Producing new enterprise services

To make the relationship of inter-enterprise collaboration management and MDA process more concrete, an enterprise service production process is briefly sketched. Figure 2 illustrates the process and the flow of model information in the architecture.

For the service elements two sources of model information is needed: type repository and realization models. The type repository is used for retrieving an existing service interface definition with behavioral, syntactic and NFA related information. Naturally, the MDA process can start by definition of new service interface type or subtype, and its publication to the type repository; in the publication phase, relationships to other existing models can be stated (or generated) and verified. The realization models should be available as a repository as well; most likely the repository is embedded into MDA development environment and thus may be vendor specific although free exchange of models would be ideal.

The service types can be located either directly browsing the type repository, or by browsing the business network models first. When an appropriate business network model is found, one of the roles can be chosen and service types associated to it can be picked up. The business network models can be stored for example as enhanced, abstract BPEL4WS [16] descriptions, or in a home-brew notation for ODP enterprise language. Service descriptions can be stored for example in enhanced WSDL [18].

The service types can then be organized as interfaces, and several implementation models can be selected to create the overall PIM for the service logic. Into this basic framework, several aspects PIMs can be intertwined to include for example non-functional property management (security, QoS, trust, policy-based protection of service abuse). The resulting network of communicating objects/components/subservices/workflow has to be analyzed for its viability, and code generated. The platform model and aspect models need to be available to describe the target environment onto which code is intended to run. Part of the platform model form facilities for binding management, which has to become a standardized, abstract PIM.

When the implementation has been generated and deployed, metainformation has to be provided: service offers exported to traders describing all service interface properties, binding requirements, range of nonfunctional properties that can be adapted to, etc.

5 Conclusion

This paper tries out some initial ideas on how MDA tools could be used for production of enterprise services that are autonomous but interoperable within a collaboration environment. The environment outline is that of web-Pilarcos project.

The exercise shows that the MDA process is applicable when the tools used are able to take several input models and produce several different kind of output: code and metainformation for the runtime environment.

Essential for the produced applications is that they use the abstract services of the collaborative operational environment, especially the binding facilities. Other parts of the computing platform are fairly much isolated.

The federation contract structures that are focal in the web-Pilarcos architecture capture requirements from all ODP viewpoint models. Consequently, the MDA stepwise process running from CIM to PIM and to PSM must pick up requirements from the contract structures at each step. Likewise, requirements for information contents and presentation should follow the same method.

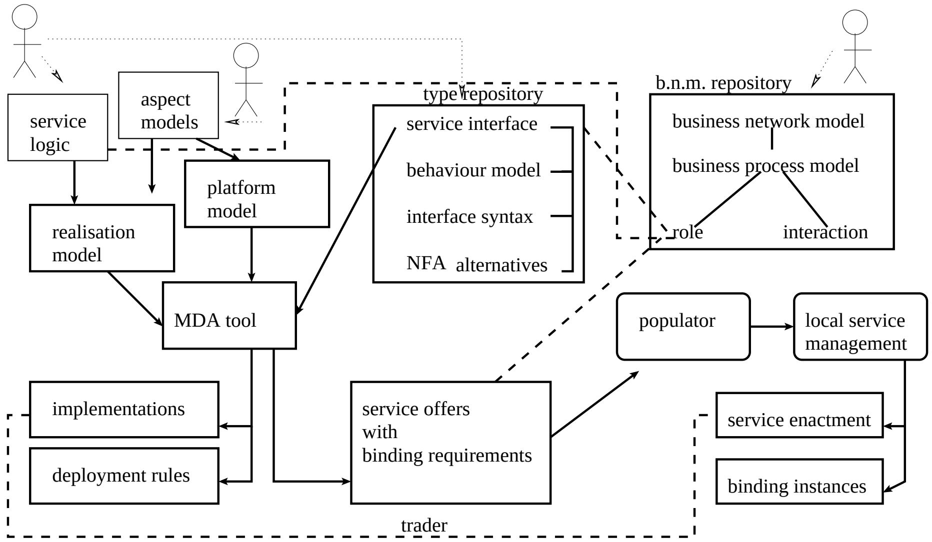

Figure 2. Flow of models.

Using the collaborative business network models as a source of requirements for the enterprise applications cause the need of identifying some commonly accepted property, policy, and behaviour alternatives. The ontologies for these should be presented within the type and model repositories of the collaboration infrastructure; this provides a method for disseminating standard ontologies. It is not plausible to develop a unified ontology for all services, but instead, it is probable that certain ontologies can have a business network model or a few business domain as their scope.

6 Acknowledgment

This workshop paper stems from work performed in the web-Pilarcos project at the Department of Computer Science at the University of Helsinki. In web-Pilarcos, active partners have been VTT, Elisa and SysOpen. The web-Pilarcos< project is a member in national ELO program (E-Business Logistics) [15].

References

[1] *Model Driven Architecture (MDA)*. ormsc/01-07-01.

[2] *MDA Guide Version 1.01.*, 2003. omg/2003-06-01.

[3] Rosettanet implementation framework: Core specifi cation v02.00.00, 2004. http://www.rosettanet.org/.

[4] J. Bezivin, S. Gerard, P.-A. Muller, and L. Rioux. Mda components: Challenges and opportunities. In *Metamodelling for MDA*, University of York, 2003.

[5] L. M. Camarnha-Matos. Infrastructure for virtual organizations – where we are. In *Proceedings of ETFA'03 - 9th international conference on Emerging Technologies and Factory Automation*, Lisboa, Portual, Sept. 2003.

[6] D. S. Frankel. *Model Driven Architecture - Applying MDA to Enterprise Computing*. OMG Press, 2003.

[7] A. Gavras, M. Belaunde, L. F. Pires, and J. P. A. Almeira. Towards an mda-based development methodology for distributed applications. In *First European Workshop on Model Driven Architecture with Emphasis on Industrial Application*, 2004.

[8] ISO/IEC JTC1. *Information Technology – Open Systems Interconnection, Data Management and Open Distributed Processing. Reference Model of Open Distributed Processing. Part 3: Architecture*, 1996. IS10746-3.

[9] L. Kutvonen. Controlling dynamic ecommunities: Developing federated interoperability infrastructure. In *INTEREST 2004 workshop*.

[10] L. Kutvonen. Using business network models in web-pilarcos. In *EMOI 2004 workshop*.

[11] L. Kutvonen. Automated management of interorganisational applciations. In *EDOC2002*, 2002.

[12] L. Kutvonen. B2b middleware for managing process-aware ecommunities. In *submitted manuscript*, 2004.

[13] D. S. Linthicum. *B2B Application Integration - eBusiness-Enable Your Enterprise*. 2001.

[14] T. Ruokolainen. Component interoperability. Master's thesis, University of Helsinki, Department of Computer Science. In Finnish. Manuscript to be accepted in April 2004.

[15] TEKES. *ELO program*, 2003. http://www.tekes.fi /programs/elo.

[16] S. Thatte. Business process execution language for web services, version 1.0. Technical report, July 2002. http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/.

[17] M. Vähäaho. Arkkitehtuurikuvauksia hyödyntävä meklaus. Master's thesis, Department of Computer Science, University of Helsinki, Dec. 2002. C-2003-NN.

[18] WSDL Specifi cation. Technical report, 2004.