Unified Reversible Life Cycle for Future Interoperable Enterprise Distributed Information Systems

Zhiying Tu, Gregory Zacharewicz, David Chen

Unified Reversible Life Cycle for Future Interoperable Enterprise Distributed Information Systems

Z. Tu, G. Zacharewicz and D. Chen
IMS-LAPS / GRAI, Université de Bordeaux – CNRS, 351, Cours de la Libération, 33405, TALENCE cedex, France

Abstract. This paper aims at improving the re-implementation of existing information systems when they are called to be involved in a system of systems, i.e. a federation of enterprise information systems that interoperate. The idea is reusing the local experiences coming from the development of the original information system with the process of Model Discovery and Ontological approach. We give first, a review of ongoing researches on Enterprise Interoperability. The MDA can help to transform concepts and models from the conceptual level to the implementation. The HLA standard, initially designed for military M&S purpose, can be transposed for enterprise interoperability at the implementation level, reusing the years of experiences in distributed systems. From these postulates, we propose a MDA/HLA lifecycle to implement distributed enterprise models from the conceptual level of federated enterprise interoperability approach. In addition to this classical development, we propose a model reversal methodology to help re-implement the legacy information system, in order to achieve the interoperability with other systems.

Keywords: Interoperabilitty, HLA, MDA, FEDEP, Information System

1 Introduction

In the globalised economic context, the competitiveness of an enterprise depends not only on its internal productivity and performance, but also on its ability to collaborate with others. This necessity led to the development of a new concept called interoperability that allows improving collaborations between enterprises. No doubt, in such context where more and more networked enterprises are developed; enterprise interoperability is seen as a more suitable solution to total enterprise integration.

Since the beginning of 2000, several European research projects have been launched to develop enterprise interoperability (IDEAS, ATHENA, INTEROP). Three main research themes or domains that address interoperability issues were

identified, namely: (1) Enterprise modeling (EM) dealing with the representation of the inter-networked organization to establish interoperability requirements; (2) Architecture & Platform (A&P) defining the implementation solution to achieve interoperability; (3) Ontologies (ON) addressing the semantics necessary to assure interoperability.

This paper intends to propose an improvement of the re-implementation of existing information systems when they are called to be involved in a system of systems. Compare to the traditional IS and Simulation development process, we involve the bottom-up process of Model Discovery and Ontological approach into our framework. In the framework, first, we align MDA and HLA to propose a lifecycle to implement distributed enterprise models from the conceptual level of federated enterprise interoperability approach. Besides that, a model reversal methodology is integrated with the MDA/HLA development lifecycle to assist re-engineer the legacy information system, in order to achieve interoperability with other systems. Compared to other techniques relevant to interoperability problems, such as SOA, our methodology focuses on providing a standard service API for all the participants. They can use this API to develop communication agent which adapts to the existing IT system without changing it.

The paper is structured as follows. We start out with a survey of related work in the area of HLA, MDA and their harmonization, as well as in the area of reengineering to model-driven architectures (section 2). Then, we describe our motivation of this work (section 3). After that, we explain two main parts of our methodology, alignment of MDA and HLA FEDEP (section 4) and Model Reversal (section 5).

2 IS and Simulation Development Life Cycle: State-of-the-Art

2.1 HLA

The High Level Architecture (HLA) is a software architecture specification that defines how to create a global software execution composed of distributed simulations and software applications. This standard was originally introduced by the Defense Modeling and Simulation Office (DMSO) of the US Department Of Defense (DOD). The original goal was reuse and interoperability of military applications, simulations and sensors.

In HLA, every participating application is called federate. A federate interacts with other federates within a HLA federation, which is in fact a group of federates. The HLA set of definitions brought about the creation of the standard 1.3 in 1996, which evolved to HLA 1516 in 2000 [1].

The interface specification of HLA describes how to communicate within the federation through the implementation of HLA specification: the Run Time Infrastructure (RTI).

Federates interact using services proposed by the RTI. They can notably “Publish” to inform about an intention to send information to the federation and “Subscribe” to reflect some information created and updated by other federates. The information exchanged in HLA is represented in the form of classical object

class oriented programming. The two kinds of object exchanged in HLA are Object Class and Interaction Class. Object class contains object-oriented data shared in the federation that persists during the run time; Interaction class data are just sent and received information between federates. These objects are implemented within XML format. More details on RTI services and information distributed in HLA are presented in [1].

The FEDEP (Federation Development and Execution Process) describes a high-level framework for the development and execution of HLA federation. FEDEP uses the seven-step process to guide the spiral development of the simulation system through phases of requirements, conceptual modelling, design, software development, integration, and execution [2].

2.2 MDA

The first methodology studied is Model Driven Architecture (MDA). This methodology defined and adopted by the Object Management Group (OMG) in 2001, (updated in [3]) is designed to promote the use of models and their transformations to consider and implement different systems. It is based on an architecture defining four levels, which goes from general considerations to specific ones.

- CIM Level (Computation Independent Model): focusing on the whole system and its environment. It is also named « domain model », it describes all work field models (functional, organisational, decisional, process…) of the system with an independent vision from implementation.
- PIM Level (Platform Independent Model): modelling the sub-set of the system that will be implemented.
- PSM Level (Platform Specific Model): that takes into account the specificities related to the development platform.
- Coding Level: The last level consists in coding enterprises applications (ESA: Enterprise Software Application).

To complete this description, a Platform Description Model used for the transformation between PIM level and PSM level is added to these four kinds of models corresponding to four abstraction levels.

2.3 MDA/HLA harmonization

In [4], Andreas Tolk mentions that while the HLA can help the MDA to improve in its principles concerning distributed simulation systems, the MDA can help the HLA implementers to improve their products based on the experiences of the OMG partners and the related software industry. Meanwhile, [4] proposes to consider five aspects for merge HLA and MDA together: Domain Facilities, Pervasive Services, RTI as Middleware, Federation Development Tools, and Data Engineering.

A solution of applying MDA to HLA is proposed in [5]. In this methodology, the Component-based development (CBD) is underlined. The design goals and

philosophy of CBD also share many similarities with the goals of HLA federate development, including the promotion of reuse and interoperability. However, HLA currently has no standardized development approach to CBD, known as a component model, analogous to the CORBA Component Model (CCM) or the Enterprise Java Bean (EJB) container. As a result, there is no commonality in design and implementation between federates, nor any formal way of separating and reusing the behavioural aspects of an HLA component. This results in an increase in development and maintenance costs, as well as impacting on the potential for reuse outside of the FOM for which the federate was built. As the result, they introduce SCM (simulation component model) into HLA (fig. 2.1.). SCM is one potential candidate for the definition of a standardized model for component-based simulation development. This architecture employs a component model based on the OMG's CCM, and describes how the separation of integration logic and simulation behaviour can be achieved for an HLA federate. This SCM development model also provides a commonality of design between federates, allowing them to access both HLA and extended CBD services (such as data transformation services between federates and the FOM) in a consistent manner.

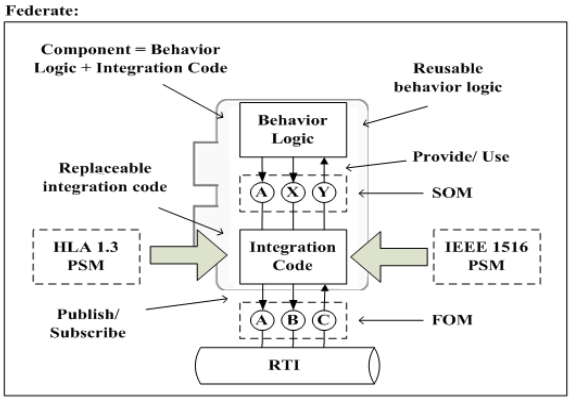

Fig. 2.1. simulation component model

2.4 MDA/FEDEP

As mentioned in [6], FEDEP and MDA are successful within their particular community. FEDEP and MDA are separate, but alignable development life cycle. Each follows basic systems engineering process through analysis is done in the following areas:

- Definition of requirements
- Definition of attributes and behaviors through a functional analysis of the requirements
- Narrative and graphical Expression of requirements, functions and analysis
- Design of the 'product'

Based on the FEDEP MDA alignment shown in fig. 2.2, [6] also proposes a development lifecycle as illustrated in fig. 2.3, which surrounds the testing phase, in order to implement the VV&A.

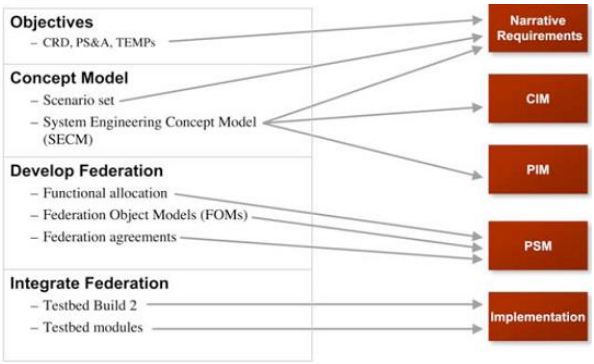

Fig. 2.2. FEDEP MDA alignment

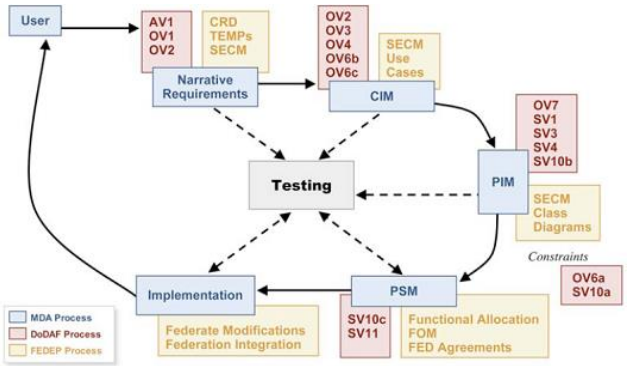

Fig. 2.3. FEDEP MDA alignment based lifecycle

2.5 MDA-Based Reverse Engineering

A framework to reverse engineering MDA models from object oriented code is presented in [7]. This framework is based on the integration of compiler techniques, metamodeling and formal specification and it also distinguishes three different abstraction levels linked to models, metamodels and formal specifications, as fig. 2.4 shows.

In this framework, the model transformations are based on classical compiler construction techniques at model level. At the metamodel level, MOF metamodels are used to describe the transformations from model level. MOF metamodels describe families of ISMs, PSMs and PIMs. Every ISM, PSM and PIM conforms to a MOF metamodel. Metamodel transformations are specified as OCL contracts between a source metamodel and a target metamodel. MOF metamodels “control”

the consistency of these transformations. The level of formal specification includes specifications of MOF metamodels and metamodel transformations in the metamodeling language NEREUS that can be used to connect them with different formal and programming languages. The transformations are based on static and dynamic analysis at model level [7].

- Static analysis extracts static information that describes the software structure reflected in the software documentation (e.g., the text of the source code). Static information can be extracted by using techniques and tools based on compiler techniques such as parsing and data flow algorithms.
- Dynamic analysis extracts dynamic analysis information, which describes the structure of the run-behavioral, by using debuggers, event recorders and general tracer tools. Dynamic analysis is based on an execution model including the following components: a set of objects, a set of attributes for each object, a location and value of an object type for each object, and a set of messages. Additionally, types such as Integer, String, Real and Boolean are available for describing types of attributes and parameters of methods or constructors.

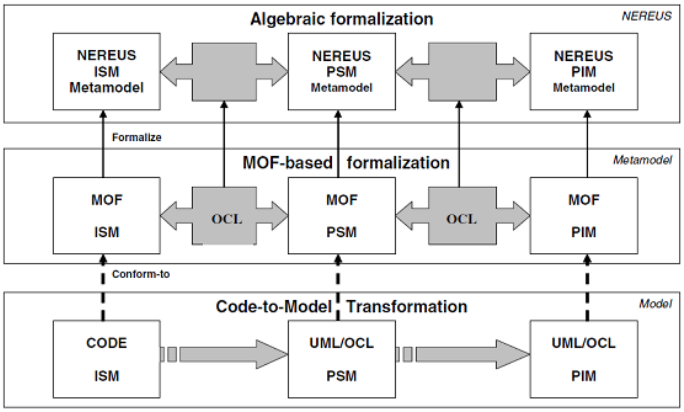

Fig. 2.4. MDA-based reverse engineering framework

3 Studied Context

A multiple agent/HLA enterprise interoperability methodology is proposed in [9]. It mentions several enterprises who all participate in a cooperative project, and they need to exchange various information any time, so the framework (fig. 3.1) defined in the methodology provides a platform based on HLA, where each enterprise plays as a federate and connect to each other.

This methodology will involve MDA&HLA FEDEP alignment in order to use both of their advantages to realize proper component reuse and rapid development.

Besides that, in order to achieve a rapid federate development according to the legacy IT system, the model reverse engineering is integrated with this alignment.

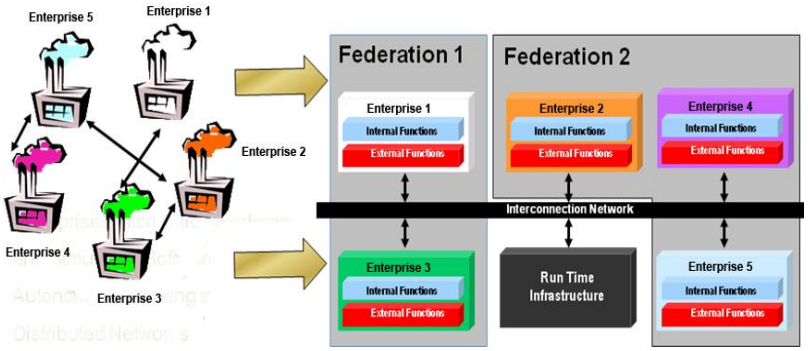

Fig. 3.1. Interoperable System of Information-Systems

4 Alignment of MDA and HLA FEDEP

In this section, we propose a new development lifecycle to reconstruct HLA FEDEP and MDA, and generate a new five steps development framework (as shown in fig. 3.2). This new methodology aims to adopt the strong points from both HLA FEDEP and MDA while overcoming their weak points, then, to achieve proper component reuse and rapid development.

4.1 Phase 1: Domain requirment definition

The main task of this phase is to collect clear and enough requirements from customer in order to define the objective of the system, to describe the environment of the system, the senario of the system. At the same time, all these definition and description need to be reasonable, understandable for all the stakeholder.

CIM of MDA has more similar task with Define Federation Objectives, Develop Federation scenario together in HLA FEDEP. As the result, we align them in this phase, to convert the user requirement, which is more textual based, into more visual model, such as UML use case to derive the federation requirement, etc.

4.2 Phase 2: Domain scenario systematization

The main task of this phase is to refine the domain scenario and business process defined in the first phase, to identify and describe the entities involved in the scenario and business process. And then, to define the relationships among entities and behaviors, events for each entity, etc.

This phase integrates PIM in MDA, which describes the operation of system but doesn't address the detail platform information yet, as well as steps of Perform Conceptual Analysis, Develop Federation Requirements and Select Federates in HLA FEDEP, which also define and select general participators of the federation, then describe their relationship, behaviours and event in general.

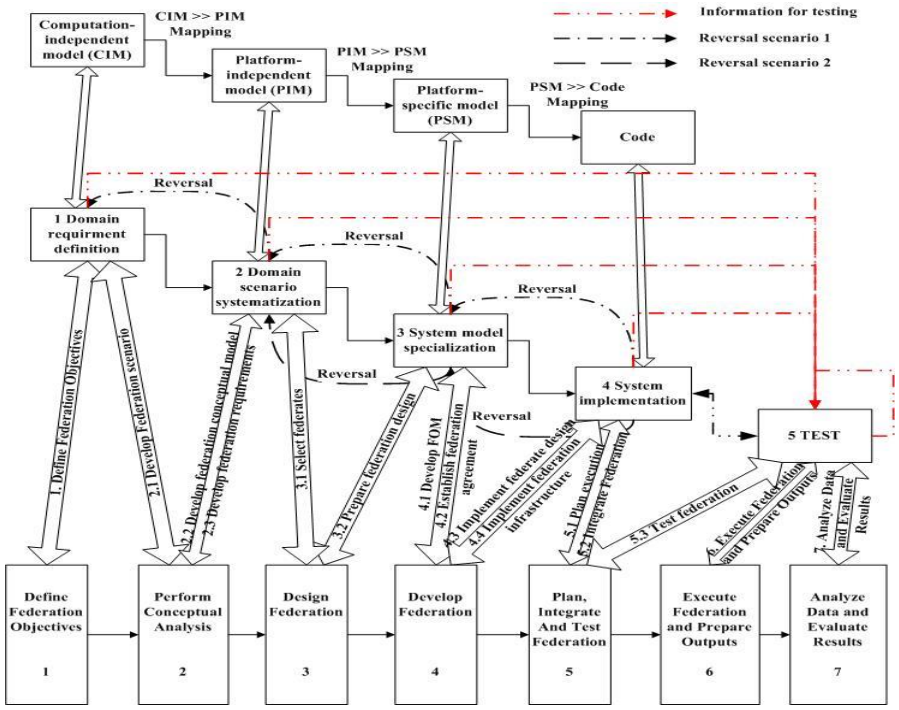

Fig. 3.2. IS and Simulation Devellopment Life Cycle

4.3 Phase 3: System model specialization

In this phase, according to the technique chosen and platform selected, the system needs to be refined, for instance, to refine federation and federate structure, to allocate functions and attributes, etc. Detailed design will carry out at this time.

This phase integrates the following parts in MDA and FEDEP. PSM in MDA, which is in the form of software and hardware manuals or even in an architect's head, will be based on detailed platform models, for example, models expressed in UML and OCL, or UML, and stored in a MOF compliant repository. The Prepare federation design, Prepare plan, Develop FOM, and Establish federation agreement in FEDEP will produce federate responsibilities, federation architecture, supporting tools, integration plan, VV&A plan, FOM, FED/FDD and time management, date management, distribution agreements, etc.

4.4 Phase 4: System Implementation

This phase's task is to transfer the specific system model into code, to create the executable federation and runable federate.

At this level, MDA has various transformation techniques from model to code. In the FEDEP, Implement Federate designs will provide modified and/or new

federates and supporting database. Implement Federation Infrastructure will provide implemented federation infrastructure and modified RTI initialization data. Plan Execution and Integrate Federation will provide execution environment description and integrated federation.

4.5 Phase 5: Test

Throughout the previous steps of the MDA and HLA FEDEP alignment process, testing is essential to ensure fidelity of the models. Testing phase includes the Test Federation, Execute Federation and Prepare Outputs, and Analyze Data and Evaluate Results in HLA FEDEP. Meanwhile, it will also refer to the outputs from the previos steps, such as the original user requirment in the first step, and federation test criteria from second phase.

5 Model Reversal

This section describes a brand-new process of model reverse engineering with different senarios constraints (see fig. 3.2). The reverse process will re-characterize the legacy system in order to capitalize on the information and functions of the existing system, and make it easy for reusing in a new HLA compliant system. This methodology will assite to HLA FEDEP / MDA alignment mentioned in previous section, to fully achieve rapid development of federation and/or federate based on the legacy IT systems. We distiguish two kinds of reversal scenarios as following.

1. First, when an enterprise intents to start exchanging information in a new cooperative project with other enterprises. In that case, the HLA federation has not been created yet, so we propose to reverse the code of the legacy information systems to the first definition phase (domain requirment definition). Then from top to down, we generate the model for each phase, finally we produce a federation and federate rapid development template.
2. Second, if an enterprise intents to paticipate in a existing cooperative project and exchange data with other heterogeneous IS. Thus, we assume an HLA federation has already been created. Here, according to the HLA FEDEP, federate starts to be considered from the second step (Perform Conceptual Analysis) as the reversal scenario 2 shows in fig. 3.2. Therefore it is not necessary to reverse to the first phase, the reversal can stop at the second phase (Domain scenario systematization). One will only reuse the model of the existing federation to create the model for the federate related to the legacy system of the new participator. Finally, the model of the existing federation and the new federate model are used to generate the code template for the new federate for rapid development.

6 Conclusion and Future Works

Based on the state-of-the-art, we have proposed a new systematic methodology, which is a valuable outcome of the combination of HLA FEDEP & MDA alignment and Model reverse engineering. This methodology provides a new five steps process to develop models of simulation starting from conceptual enterprise models. In addition, it also bridges the gap from concepts to implementation in the field of enterprise modelling by offering a new standardised and reversible approach. This methodology seems promising regarding to real enterprise information system requirement of distribution, federated interoperability and agility of adapt to dynamic context.

Compared with other techniques, which can solve the interoperability problem, such as SOA, our methodology is trying to provide a standard service API for all the participants, who can use this API to develop communication agent which adapts to the existing IT system without changing it.

Up to now, this work is still in a research process. The methodology presented (each phase of HLA/MDA alignment and the model reversal process) still needs to be refined and detailed. A case study will allow testing the proposed approach in an industrial context.

7 References

[1] IEEE std 1516.2-2000. IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Federate Interface Specification, Institute of Electrical and Electronic Engineers.

[2] IEEE std 1516.3-2003. IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA) - Federation Development and Execution Process (FEDEP), The Institute of Electrical and Electronic Engineer.

[3] OMG, 2003. MDA Guide Version 1.0.1. Object Management Group, Document number: OMG / 20030601 Available from: www.omg.org/docs/-omg/03-06-01.pdf [accessed 15 June 2009]

[4] Andreas T, (2002), Avoiding another Green Elephant—A Proposal for the Next Generation HLA based on the Model Driven Architecture. Proceedings of the 2002 Fall Simulation Interoperability Workshop: 02F-SIW-004

[5] Shawn P, (2003), The Next Step Applying the Model Driven Architecture to HLA. Proceedings of the 2003 Spring. Workshop: 03S-SIW -123

[6] Trbovich S, Reading R, (2005) Simulation and Software Development for Capabilities Based Warfare: An Analysis of Harmonized Systems Engineering Processes. Proceedings Spring Simulation Interoperability Workshop: 05S-SIW-106

[7] Favre L, Martinez L, Pereira C, (2008), MDA-Based Reverse Engineering of Object Oriented Code. SERA'08: 153-160

[8] Favre L, (2008), Formalizing MDA-based Reverse Engineering Processes. SERA'08: 153-160

[9] Zacharewicz G, Chen D, Vallespir B, (2008), HLA Supported, Federation Oriented Enterprise Interoperability, Application to Aerospace Enterprises. Proceedings of 2008 EURO International Simulation Multi-conference: 08E-SIW-074

[10] Meta Object Facility (MOF) Specification v1.4, OMG Document formal/02-04-03, http://www.omg.org/cgi-bin/apps/doc?formal/02-04-03.pdf