CHALLENGES AND PERSPECTIVES IN THE DEPLOYMENT OF DISTRIBUTED COMPONENTS-BASED SOFTWARE

Mariam Dibo and Noureddine Belkhatir
Laboratoire d'Informatique de Grenoble, 681, Rue de la Passerelle, BP 72, 38402 St Martin d'Hères, France

Keywords: Deployment, Components based software engineering, J2EE, CCM, .NET, D&C, Generic deployment system, MDA, Deployment process.

Abstract: Software deployment encompasses all post-development activities that make an application operational. It covers different activities such as packaging, installation, configuration, application start and updates. These deployment activities on large infrastructures are more and more complex leading to different works generally developed in an ad'hoc way and consequently specific to middleware such as for instance J2EE, .net or CCM. Every middleware designs specific deployment mechanisms and tools. The objective of this work is to propose a generic deployment approach independently of the target environments and to propose necessary abstractions to describe the software to be deployed, the deployment infrastructures and the deployment process with the identification and the organization of the activities to be carried out and the support for its execution. Our approach is model driven and our contribution is about a generic deployment framework.

1 INTRODUCTION

An important issue of component based software engineering is the deployment of components in decentralised locations in an efficient, safe and consistent manner. The deployment life cycle encompasses all the post-development activities of an application which makes the software useful. It is an important step in software life cycle, which for a long time has been reduced to only installation. Today, the components approach and the distribution make deployment complex. Many deployment tools exist but they are often built in an ad'hoc way, specific to a technology or to an architecture and covering partially the deployment life cycle. This paper reviews this important domain of software life cycle, emphasizing the pros and cons of each deployment approach.

The challenge is to develop a generic framework encompassing specific tools and supporting the whole deployment process.

The rest of this paper is organized as follows: Part 2 presents related works. We split this part in two sections. In the first section, we present approaches developed by the industrials leading to tools and platforms developed in an ad' hoc manner.

The second section aims at specifying high level abstraction approaches using models, meta-models and their transformations. Examples of these approaches are Software Dock (Hall et al., 1999), SOFA (Bures et al., 2006), UDCM (Hnetynka, 2005) and Dance (Edwards et al., 2004). We put in this category the works developed by the OMG group identified as D&C (OMG, 2006b) and based on a model-driven approach.

In the last part we present briefly our approach highlighting the concepts and the architecture of the generic deployment framework we propose.

2 RELATED WORKS

2.1 Industrial Approaches

In this category, most of the existing technology of deployment is built in an ad' hoc manner. Therefore, every system has its own tool or its own method of deployment covering partially the deployment life cycle. In this category, we include industrial proposals such as CCM (OMG, 2006a), EJB (SUN, 2009], OSGI (Alliance, 2003) or .NET (Lowy, 2001). These systems integrate some preoccupations such as the persistence or safety but supply solutions

Dibo M. and Belkhatir N. (2009).
CHALLENGES AND PERSPECTIVES IN THE DEPLOYMENT OF DISTRIBUTED COMPONENTS-BASED SOFTWARE.
In *Proceedings of the 11th International Conference on Enterprise Information Systems - Information Systems Analysis and Specification*, pages 403-406
DOI: 10.5220/0002009904030406

which are too ad-hoc. For example the composition of units is still not taken into account, the programming language choice is very binding, and communication is very dependent on the context. A general weakness is the lack of abstraction allowing a better re-use of components. For lack of a frame of composition in the industrial models, it is the Glue code which assures the communication between components if they are distributed.

2.2 Academic Approaches

Model based systems such as D&C (OMG, 2006b), UDCM (Hnetynka, 2005) provide expressive abstractions to control deployment. These systems enhance a technology transition based on models and meta-models offering more generality and elevating the level of abstraction. The key capabilities are:

- Identification of the common services separation of concerns with clear identifications of the different models (applications, locations, deployment planner and orchestration).
- Mapping based on MDA (OMG, 2005) a transformational approach to the underlying middleware.
- Automatic deployment and life cycle management.

3 OUR APPROACH OVERVIEW

We know unquestionably that the concept of architecture with components consists in creating reusable entities and in developing personalized software thanks to the appropriate assembly of these components. In such a context, the roles of components developers and applications composers become clearly different. So, components developers work out on generic components whereas the applications composers concentrate on the application domain in assembling and in configuring generic components available in business. Thus, a component adheres to a component model, which establishes the standard for the implementation and the interoperability of the component. In such an environment, where the development of components tends to be more and more independent from their re-use, it is necessary to have a deployment machine which will allow assembly and to distribute applications correctly with components whatever their implementation may be. Figure 1 below represents the deployment process of components-based software which is constituted by several activities in correlation. Thus, deploying a component based software consists in distributing components in specific locations and in managing the constraints of placement, dependencies and configuration. Once deployed, a software system is available for use.

We propose a generic deployment framework. By generic we understand, the fact of being able to deploy any application regardless of the implementation technology.

(1) Application modeling, concerns the description of software architecture. It allows to model for every application, the various components which constitute it. For every composite or primitive component, we can specify its assembly, its constraints of dependences as well as its software and material needs.

(2) Network domain modeling, concerns the description of the locations network. It allows to model for every domain, the various locations it contains. For every location, we can specify the offered material and software resources.

(3) Enterprise thesaurus modeling, allows to model the similarity between the concepts used in a specific enterprise or in a business domain.

(4) Strategy modeling, allows defining the strategy used to deploy applications.

(5) In **deployment processes** many concepts (application, domain, enterprise thesaurus and strategy) are manipulated. It processes components **placements**. It generates and assures the **deployment plan** consistency and supplies the specific **deployment descriptor.** It **executes the plan** following specifics strategies to the execution environment.

Placement is the association of an application component and a domain node. A placement (Ci, Nj) is valid if and only if all component Ci constraints are satisfied by node Nj resources.

Deployment Plan, for an application A composed of component C1 to Ci where i> 1 and for a domain D formed from locations L1 to Lj where j> 1, the deployment plan is all the valid placements (Ci, Lj). The placements are fulfilled by the deployment planner which contains all data and all necessary strategies to make the mapping viable.

The deployment plan (PIM), will be instantiated generating in the **deployment descriptor** specific to the execution platform (PSM), for instance, in compliance with the EJB platform.

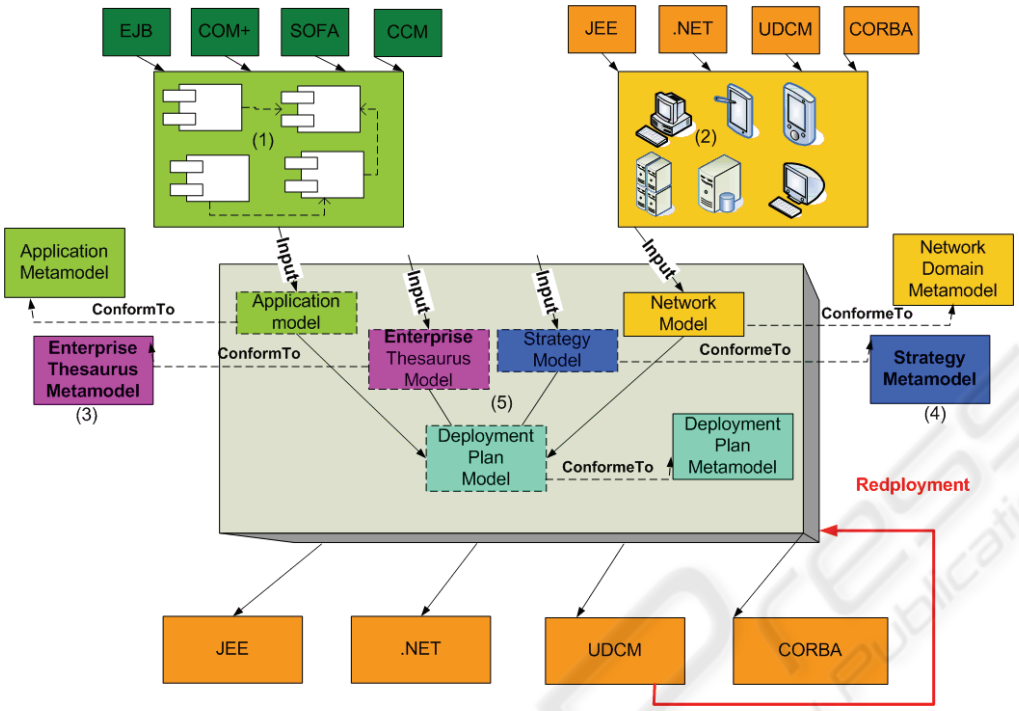

Figure 1: Generic deployment engine.

The plan execution is the orchestration of the plan in a defined order. As for instance download, install, activate and reinstall. Our team has developed an environment for dynamic reinstallation and adaptation DYVA (Ketfi et al., 2002). DYVA covers a part of plan execution.

4 CONCLUSIONS

Deployment becomes more and more complex when deploying large systems on large infrastructures.
On the one hand, any ad' hoc solutions for deploying monolithic or component based systems exist.
On the other hand, there is a new approach to deployment technology. In recent years there have been many works in development by academics focusing on a new generation of systems. These approaches enhance a technology transition. They have shown the potential of using a model-driven approach such as MDA.
The defined models are based on expressive and simple abstractions so the application, the location, the deployment process and its orchestration can be built on top of that common foundation.
We hope that the deployment framework architecture we propose is a contribution to this new generation of systems.

At the moment we implemented the module of application modeling, the module of network domain modeling and the protocol of plan generation by taking into account the strategy by default. In the future works we plan to add a strategies specification language more elaborated and benefiting from various approaches.

REFERENCES

Alliance, OSGI Service Platform, Release 3, IOS Press, Inc., 2003.

Bures, T., Hnetynka, P., Plasil, F., SOFA 2.0: Balancing Advanced Features in a Hierarchical Component Model, SERA'06, Fourth International Conference on Software Engineering Research, Management and Applications 09-11 Aug. 2006, p. 40 – 48.

Edwards, G., Deng, G., Schmidt, D. S., Gokhale, A., Natarajan, B., Model-driven Configuration and Deployment of Component Middleware Publisher /Subscriber Services, Proceedings of the 3rd ACM International Conference on Generative Programming and Component Engineering, Vancouver, CA, October 2004.

Hall, R. S., Heimbigner, D., Wolf, A. L., A cooperative approach to support software deployment using the software dock, Proceedings of the 21st international conference on Software engineering, p.174-183, May 16-22, 1999, Los Angeles, California, United States.

Hnetynka, P., A model-driven environment for component deployment, Software Engineering Research, Management and Applications, 2005. Third ACIS International Conference on 11-13 Aug. 2005, p. 6 – 13.

Ketfi, A., Belkhatir, N., Cunin, P.Y, Adaptation Dynamique Concepts et Expérimentations Proceedings of the 15th International Conference on Software & Systems Engineering and their Applications ICSSEA'02, Paris, France, December 2002.

Lowy, J., COM and .NET component services, O'Reilly & Associates, 2001.

OMG, CORBA Component Model Specification http://www.omg.org/docs/formal/06-04-01.pdf, Version 4.0, Technical Report, April 2006.

OMG, Deployment and Configuration of Component-based Distributed Application Specification, Version 4.0, Technical Report, April 2006.

OMG, Model Driven Architecture, OMG Document ormsc/05-04-01, 2005.

SUN, Enterprise Java Bean, Adress = http://java.sun.com, Path = /products/ejb/.