A Practical Experience on Model-driven Heterogeneous Systems Integration

Antonio Estévez[1], José D. García[1], Javier Padrón[1], Carlos López[1], Marko Txopitea[2], Beatriz Alustiza[3], José L. Roda[4]

[1]Open Canarias, SL, Elías Ramos González, 4, ofc. 304, S/C de Tenerife, 38001 España

[2]Open Norte, S.L., Madariaga Etorbidea, 1 – 4. Ezkerra, 48014 Bilbao, España

[3]IZFE , S.A., Pinares Plaza, 1 – 4. solairua, 20001 Donostia – San Sebastián, España

[4]ULL, Escuela Técnica Superior de Ingeniería Informática
Universidad de La Laguna, La Laguna, España

Abstract. The integration of heterogeneous systems is usually a complex task. In this study we present a strategy which can be followed for the integration of a framework based on Struts and J2EE, the transactional system CICS and the document manager FileNet. The principal aim of the project was to redefine the work methodology of the developers in order to improve productivity. Following model-based development strategies, especially MDA, a single framework for the three environments has been developed. Independent metamodels were created for each one of the environments, which finally led to a flexible, open and unified metamodel. The developer could then increase his productivity by abstracting from the particular implementation details related to each environment, and putting his efforts in creating a business model that is able to represent the new system.

1 Introduction

Most large business corporations and concerns use frameworks and heterogeneous tools in the running of their systems. Most of these systems need to integrate several architectures, technologies and information systems. At the moment, there are few consolidated solutions to solve these problems at a reasonable cost, which as well as being in house problems, will also depend on the technologies used to solve them.

The latest leanings have been towards strategies based on the Model-Driven Software Development (MDSD [13]), and more especially MDA [12] as a possible solution to most of the existing problems. Through MDA strategies, businesses can make sure that their business plans remain valid, independent from the frequent changes in the technology involved. Examples of the use of these kinds of strategies can be found in the following references: [3], [8].

In this paper we describe a general methodology for the integration of complex systems, based on the fundamental principles of MDSD, and applied in a real corporative environment. The principal objective of this project has been to put into

Gaillard G. (2006).
Model-Driven ERP Implementation.
In *Proceedings of the 2nd International Workshop on Model-Driven Enterprise Information Systems*, pages 98-107
DOI: 10.5220/0002482100980107

place a system of high productivity in order to develop J2EE [7] applications, which can interoperate with transactional systems such as CICS [4] and with content managers such as FileNet [5].

In part 2, the technological area is described along with the different platforms that have to be integrated. Then, we outline the development of three practical cases, using the selected platforms. The results of the application of the methodology, conclusions, and areas of future study will conclude this work.

2 Platforms Used in the Project

The Foral Society for Information Technology, pertaining to the Foral Department of Gipuzkoa (IZFE) has established and maintain an IT zone, with machines and servers made up from an IBM mainframe as well as more than 130 Windows, Unix and GNU/Linux servers, which are utilised by the Foral Department as well as the Town Councils of Gipuzkoa. IZFE is responsible for more than 90 new developments each year and at the moment has more than 300 applications up and running in a state of permanent evolution, with users as diverse as the Tax Office, the Departments of Transport, Culture and Youth, the Social Services departments, the Emergency services, as well as the Innovation. The number of persons working directly on these development projects has reached 165, without counting those collaborating within the closed environment of suppliers and providers.

The technologies used by IZFE in relation to this study are principally the Websphere Application Server for z/OS, version 5.1. The related database is DB2 Server for z/OS, version 7.1.0.

Of the technologies that we would like to integrate in the project, we would include the file manager FileNet, version 3.0, the transactional server CICS Transaction Server for z/OS, version 2.3 and the IZFE framework based on Struts [17] which allows for the development of J2EE applications.

This varied and complex group of platforms conform to the ideal scenario for the development of this project and we can apply the methodology and different integration strategies for the development of efficient software.

3 Methodology Used

Given the variety and complexity of the surroundings in which we worked, it was decided to use a bottom up methodology, beginning with the most specific aspects leading to generalizations and aspects in common. We thus planned a series of repeated tasks, differentiated by the technologies used in the study. At the end of these tasks, we proceeded to the integration of the different technologies, so as to link together with a usable common integrated model. Figure 1 shows the sequence of these tasks as well as the activities involved in each of them.

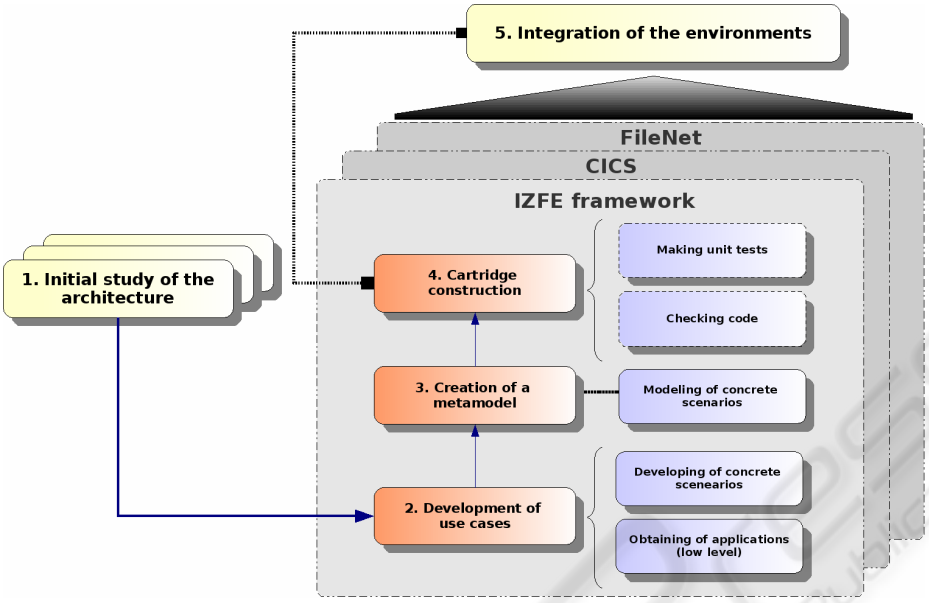

Fig. 1. Arrangement of the tasks with the methodology used.

Task 1. Initial study of the technology and architecture.
At this stage all the available information is obtained and studied. In order to do this, the IZFE is asked for all the relative information concerning his technological set-up that pertains to this project. All this information is checked, validated and developed. As far as possible, we try to make the study as near as possible to real life circumstances.

Task 2. Development of use cases.
At this stage, different applications (codes) are obtained for analysis and to define the functions required. As a result, a series of concrete scenarios should be developed, which comply in the most part with the functional requirements which have been analysed. To obtain programmes, we need wide ranging programmes to cover the technology that has been put in. These applications should for the most part cover the intrinsic necessities of the IZFE, covering the most common work tasks. The programmes are analysed and verified at a trial stage (with those technologies that already exist). These programmes are considered separately and recodified in order to easily create a metamodel, identifying structures and components which have the possibility of being considered separately, along with general concepts which previously had been modelled through the creation of templates. These templates were previously used for the automatic generation of codes in subsequent phases.

Task 3. Creation of a Metamodel.
The alignment of the project with MDSD and especially with MDA also involve the concept of UML Profiles [19], a specialised mechanism which is defined as being part of the same UML. The profiles can help shape specific aspects of a software system. The basic principle for obtaining each one of these profiles is to ascertain

generalisations between different programme languages, platforms and technologies, as well as to incorporate other relevant aspects related to the integration of inherited systems and applications.

Task 4. Cartridge Construction:
Having defined the functions and the metamodel, we can begin to construct the cartridge, whose function is to direct the working, compilation and packaging of the model exported in XMI [20]. In essence, a cartridge links the implementation of the UML profiles in a platform context with the programme language in which the code is generated, which in our case will be Java. This cartridge will contain a description where the profiles of each of the stereotypes are defined, and the corresponding template assigned. The new application is then checked through the IZFE's own technology.

These 4 tasks have been developed for the IZFE framework, CICS and FileNet areas, which we shall now explain in detail, and outline the cases where we had to customise the system in question.

4 Applying the Methodology

The development of the project followed a sequence through different technological environments. The IZFE framework was the first, as it was already identified as a key factor for the success of the project, as well as for its high level of complexity. The IZFE framework is a J2EE framework which runs on a Websphere Application Server. This is a server which is used extensively in the development of corporative web applications. The second environment considered was a transactional manager, identified as a CICS environment. In this environment there existed inherited processes and logic at a corporative level with a high strategic value. The third environment dealt with the development of an in company file manager, which was FileNet environment. In this environment there was a great quantity of high critical content, used in some areas of IZFE.

Each environment had definite tasks applied to them, using the methodology previously explained. The final phase of the project was to make a big effort to integrate all the environments into a common integrated metamodel.

4.1 IZFE Framework

Task 1. Study of the architecture. The IZFE framework is used for the creation of applications in a corporative business environment. Basically it is a fork of the Struts framework in the 1.1 version. The IZFE framework is divided into a series of subsystems, with the listener, control and presentation subsystems being of the first importance, as well as the business and the special security subsystems relevant in the corporative environment. Once the guides and reference information had been studied, an environment similar to that of IZFE was put in place.

Task 2. The development of use cases. Two applications were selected for the administration of the framework. These applications were tested and run in a simulated environment. Having these applications as a reference, the requirements could be defined for a new application and reengineering techniques were used in its implementation. During this phase, unitary components were identified, which could be used as parametric components in the metamodel.

Task 3. Creating the metamodel. The objective of the metamodel is to create a system with simplified architecture, which meets the requirements of the IZFE framework. To reach this simplified level, the components that the framework offered were mapped out to a model more inclined towards the MVC pattern [14], in which the domains are clearly defined and focussed on the functions of self contained web applications. With this simplification, we managed to reduce the elements of the MDA architecture which should be in place in order to create a more integrated application. It also permits a better distribution of the work needed to be done in the specialised areas, dividing the knowledge between different people that made up the team. The following domains and /or layers were defined: the initialisation, view, business logic, and persistence domains.

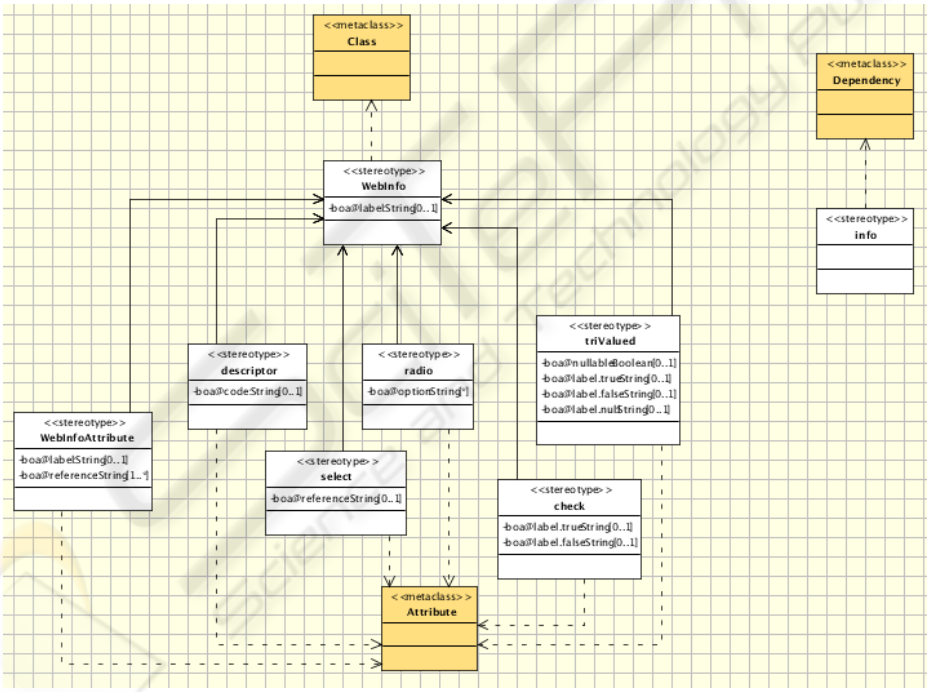

Fig. 2. Sample piece of the IZFE framework metamodel.

Task 4. Cartridge Construction. At this stage the objective proposed was the 100% code generation. This considerably increased the complexity of the problem, above all in the definition of the business logic. In order to reach this objective, state diagrams were used, incorporating into these states action semantics [1] which are described in

the specifications 1.5 of the UML. To reach this approximation, Action Specification Language was employed (ASL) [2], and with certain modifications a grammar and a parser were developed, using a compiler from the SableCC [15] compiler. In this way a cartridge which generated completely automatic applications was obtained. Now the IZFE, instead of programming these interfaces directly, uses the metamodels defined in the UML in order to represent their needs graphically. The system is capable of automatically generating codes from these diagrams.

4.2 CICS Environment

Task 1. Architecture study. The objective to be reached in this environment is the running of complex programmes hosted in CICS through J2EE components. An exhaustive study of this area was needed, due to the non-existence of any previous development programmes with the requirements specified in this study. Two key problems were identified, the communication with the EIS, and the formatting of types between domains.

Task 2. Development of existing cases. To solve the communication problem, the CICS ECI Recorder Adapter was used, put through the CTG (IBM CICS Transaction Gateway). The second problem identified in Task 1 was solved using the JRIO [10] library. Finally the minimum functions required were obtained through unitary tests in order to validate the solution.

Task 3. Creating the metamodel. The metamodel was developed by identifying the general functional components, and parametricizing the minimum information which is needed by the models in order for the previous correct generation to be attained. All this has been effected using as a reference the use cases which were put together in previous times.

Task 4. Cartridge Construction. Finally the cartridge was implemented, due to the fact that the code was automatically generated in the context of the platform, which included a small unitary test. This cartridge contains a descriptor where the profiles are defined with each of the stereotypes assigned to the corresponding templates. Beforehand, the generated systems were checked. In this way, and using a generation motor, the IZFE can describe the model graphically through simple UML diagrams, and generate 100% the code needed for the connection of the transactional manager.

4.3 FileNet Framework

Task 1. Architecture study. FileNet is a document manager with the added functions of workflow and with its own framework based on Struts. It has an API for Java which allows access to practically all of its functions. IZFE has developed and maintains a small simplified API which makes easier the running of the contents of the organisation's internal uses.

Task 2. Development of existing cases. Two extracts from two different applications were selected which made use of the API of IZFE. Based on the examples provided and using inverse reengineering, the common functions were extracted in real scenarios. Finally a series of unitary tests were made in the IZFE's environment.

Task 3. Creating the metamodel. The metamodel was developed by identifying functional components that needed to be generalised, and then parametricizing the minimum information needed to the models for their correct working. All this was carried out using past existing cases as a point of reference.

Task 4. Cartridge Creation. For the construction of the cartridge each one of the stereotypes were mapped out to the units of generation. A template was defined for each unit of generation, which allows for a generator motor for the creation of codes. The use of a cartridge allows, through the definition of UML diagrams, for the 100% generation of an access code of the resource contents defined in the corresponding document manager.

4.4 Unification of the Metamodels

The last stage defined in point 3 corresponds to then integration of the different environments. This can be done at a web level using the IZFE framework.

Initially integration with CICS was planned. A metamodel was obtained and a cartridge for the modelling independent of an actual programme, in which some entry parameters were effected and some exit parameters were obtained. Beforehand, this function was incorporated as another element, to be integrated into the metamodel used in the design of the models which are generated for the IZFE framework.

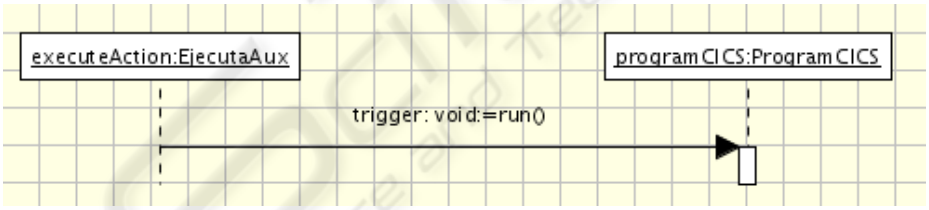

Fig. 3. Sequence diagram sample about CICS integration in IZFE framework in the view domain.

Thus, two types of integration were accommodated in the IZFE framework metamodel. One integration in the presentation domain, which permitted the use of the CICS programme using a web form. A second integration was in the business logic domain. This last integration was effected using state diagrams.

The integration with FileNet was approached in a similar way to the CICS. Once the metamodel and cartridge have been created, they can be used independently and in isolation, and be used along with the integration of the IZFE framework metamodel. This implies an enlargement, not only in the persistence domain (resource persistence), but also in the presentation domain. So a maintenance

environment has to be established, as well as the running of a general resource manager, which in turn permits in a simple form the creation, modification, cleaning and search for FileNet resources.

A two-step strategy for problem solving was used. The first step a complete system was generated using a traditional model for IZFE framework. Beforehand, a system generated to create templates was used, with a high abstract level, which in turn allowed for the definition of a series of stereotypes, which simplifies even more the definition of the integration models with FileNet. This strategy proved successful due to the few cases of variation in the initial requirements in the use of the FileNet resources.

5 Results Obtained

Using this system, an architect could construct a complete application, designing the adequate models based on UML with profiles. In order to do this, it is not necessary to be an expert in J2EE, nor in the IZFE framework, nor in CICS or FileNet; it is enough just to have a basic knowledge of these technologies and in UML.

With the correct modelling, the engine on which the project is based is capable of generating the total structure and codes needed for the start up and development of a complete and full application of a J2EE server, as in the case of a Websphere Application Server (WAS). This newly formed application is totally compatible with the corporative IZFE framework, and could be used, in the business layer with functions stored in CICS systems, or in the persistence layer, with defined resources in the document FileNet database. All this can be achieved without inserting even one line of code, and without being an expert in the technologies employed, solely by simply correctly modelling through the UML diagrams.

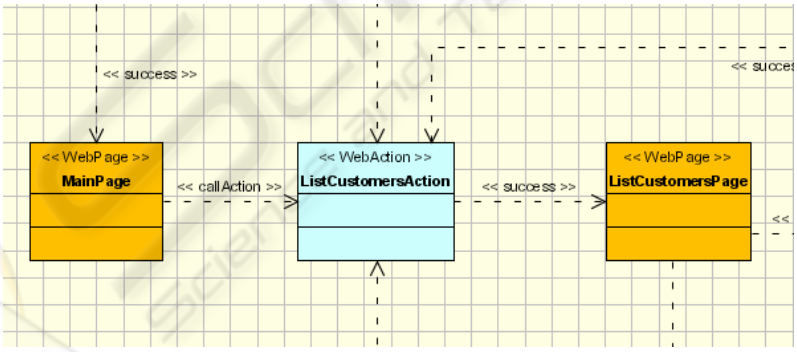

Fig. 4. Example of an actual model in the view domain.

It should also be noted that the persistence area of the business model in related databases in the IZFE framework gives a free hand for those programmers who would like to propose solutions that they feel are adequate to any given situation. We therefore propose the use of Hibernate [6] for the running of this project as a viable and efficient solution for the persistence of these objects with whichever database that

is being utilised. (in our case, DB2). This component was added, and then modelled and run through an adequate cartridge. For the connection to the EIS, in this case the CICS, we made use of the literature provided by JCA [9], where numerous references, documents and up to date texts were consulted. And for the converting of data to Cobol and vice versa, a crucial function in this area, texts and classes provided by JRIO were used. The persistence of resources was also modelled using a FileNet content manager. The metamodel had been sufficiently abstracted for it to be used generally for other content managers, by solely modifying the cartridge.

Finally, it must be noted that on the completion of this project it was possible to integrate all the previous metamodels into a common unified metamodel. With this, the complete integration of specific environments has been reached which are completely heterogeneous within a unified, efficient and ordered model, which in turn allows for the development of new systems. Anyone developing these systems would find a framework based on UML, with highly defined profiles in all three platforms, which can lead to a higher level of abstraction, working independently from the technological aspects.

6 Conclusions

This new paradigm in systems creation represents a big change in the traditional way of working of the development teams in IZFE. A new methodology was embedded as well as new work practices, along with the planning needed in the management of change to the rapid adaptation of the new paradigm to take full advantage of the new environment. The modellers of the new systems should possess a high degree of knowledge of UML in order to work on the theory and creation of these systems.
Apart from the intrinsic advantages derived from a system based on an approximation of MDA, we can also note:

- The normalisation of the systems through UML models.
- The integration of heterogeneous systems, which hide the complexities of each of the technologies in question.
- The development of one system only based on the Web.
- The considerable rise in the quality of the systems developed, given that the codes generated had been exhaustively tested.
- The possibility of a rapid development of prototypes, which could be easily converted into systems and final applications.
- The improvement in the facility of implementing the persistence of the models, independent from the continual technological change and evolution.

7 Future Proposals

For the future, we are working towards the adaptation and maintenance of the cartridge that has been made, according to how the systems already integrated have evolved (IZFE framework CICS and FileNet), as well as other technologies. Other different cartridges can be generated for other programming languages apart from

Java (.NET for example), as well as the incorporation of other tools (Spring [16]) into the corporative framework, or the interaction with other systems different from those in place, which would mean a modification of the cartridges.

There is also the possibility of integrating the technology into portlets [11], a challenge for the domain of our application. The portlet provided by Struts is recommended in order to avoid any compatibility problems with the IZFE framework controller.

In the MDA environment, it is worth noting the emergence and use of new engines for code generation. In this case, a "translational" method has been used, where, apart from the templates included in the corresponding cartridge, we also managed to put the model designed into code. There also exists at the moment other methods known as "elaborational", where changes are made to models based on QVT [18]. This method has a great future within MDA architecture.

References

1. Action Semantics Revised Final Submission. OMG document ad/01-08-04.SL
2. ASL – The Action Specification Language Reference Manual. http://www.kc.com
3. Brunton R., Brutzman D., Drake D., Hieb M., Morse K.L., Pullen J.M., Tolk A. : "Using Web Services to Integrate Heterogeneous Simulations in a Grid Environment", Lecture Notes in Computer Science, Springer-Verlag Heidelberg (2004) pp. 835 – 847.
4. CICS – Customer Information Control System. http://www-306.ibm.com/software/htp/cics/
5. FileNet http://www.filenet.com P8 3.0.0 Documentation
6. Hibernate: http://www.hibernate.org
7. J2EE – Java 2 Platform, Enterprise Edition. http://java.sun.com/javaee/index.jsp
8. Jahnke, J.H., Wadsack, J.P. : "Towards Model-Driven Middleware Maintenance", Proc. of the OOPSLA 2002 Workshop on Generative Techniques in the context of Model-Driven Architecture, Seattle, USA., November 2002.
9. JCA – J2EE Connector Architecture. http://java.sun.com/j2ee/connector/
10. JRIO – Java Record I/O. http://www-03.ibm.com/servers/eserver/zseries/software/java/jrio/overview.html
11. JSR 168, portlet specification. http://www.jcp.org/en/jsr/detail?id=168
12. MDA – Model Driven Architecture. http://www.omg.org/mda/
13. MDSD – Model-Driven Software Development. http://www.mdsd.info/
14. MVC –Model View Controller pattern. http://java.sun.com/blueprints/patterns/MVC-detailed.html
15. SableCC Parser generator. http://sablecc.org
16. Spring framework. http://www.springframework.org
17. Struts Framework http://struts.apache.org/
18. QVT – Query Views Transformations. http://www.omg.org/technology/documents/modeling_spec_catalog.htm#MOF_QVT
19. UML – Unified Modelling Language http://www.uml.org/
20. XMI – XML Metadata Interchange. http://www.omg.org/technology/documents/formal/xmi.htm