CASE STUDY

Applying MDA to the Development of Air Traffic Management Systems

CHALLENGE

The AD4 project team needed to improve both the quality and safety aspects of the development of Air-Traffic Management systems. Based on the existing D^3 System, the AD4 project was challenged with a geographically distributed development team, targeting multiple execution platforms.

WHY ENTERPRISE ARCHITECT

Sparx Systems Enterprise Architect provided the AD4 team with advanced UML 2 modeling capabilities, in addition to a powerful plug-in architecture that enabled models to be seamlessly integrated into the entire AD4 tool chain.

BENEFIT

By using a Model Driven Architecture (MDA) approach with Enterprise Architect, the AD4 team facilitated implementation for multiple target platforms, while ensuring improved system quality through early fault detection using platform independent models.

Introduction

Increased efficiency and improved quality in software development is an objective that is common to all organizations and projects. Different approaches can be adopted depending on the particularities of the organization in question and their suitability within the context of project execution and management.

Model-Driven Architecture™ (MDA) is usually portrayed as a means to perform the same amount of work with less people, ensuring in addition, improved quality of the software developed, decreased time to make changes to existing systems, and organizational know-how kept within the core of the products, namely through their models.

Some of these aspects are extremely important for specific types of systems. An example is an air-traffic management system, which must be free of defects, straightforward to modify and comprehensively documented. A typical and logical question posed by decision-making people, however, is how MDA could be successfully introduced in such projects.

We find that the three important factors for successful application of MDA are (1) basing the MDA-application on an existing product, (2) defining an appropriate life cycle, and (3) developing and using a tool chain with which to implement the software system. Applying MDA for the first time is much easier when there is available good domain knowledge and a clear idea about what should be modelled and the expected outputs from the model transformations.

The resulting improvements from the application of MDA to this software development process could be estimated through the use of metrics and empirical validation. However, at the time of writing, this exercise has not been carried out in its entirety; the results were only “validated” by customer satisfaction. With respect to the life cycle, an important prerequisite is to ensure effective and efficient learning from experience, and on-time delivery of quality results despite the potential difficulties with adopting the new technology.

This paper describes our experience when applying MDA to the development of an air-traffic management system. We discuss the complementarities of these factors using this particular case as an example.

"Enterprise Architect provides a good plug-in mechanism which allows the easy integration of custom made plug-ins. This was vital for the seamless integration within the AD4 tool chain."

—Terry Bailey
R&D Project Leader
European Software Institute

Background

Aircraft coordination in increasingly crowded airspace is becoming a major concern for air traffic management authorities around the world. Conventional management schemes are being replaced by extensively computer-integrated Air Traffic Management (ATM) Systems to maintain safety levels and increase throughput of congested airways. One of the primary goals of introducing ATM systems is to provide the controllers with as much information as they need to effectively manage the air traffic and present such information in a comprehensive form while taking care not to overload controllers with unnecessary information.

Research and development activities in this area have been established with the D^3 system, developed by NEXT S.p.A. for an Italian national project. D^3 System provides a 3D System for geo-referenced data representation and visualisation. The AD4 project extends D^3 and develops an innovative Virtual Air-Space representation for ATM Systems to provide the controllers with the ability to use 3D data about the air traffic/airport space in real time.

To develop and validate the requirements for the AD4 system, observations and in-depth analysis of the work practices and strategies used by the air traffic controllers have been carried out and a number of operational scenarios have been defined. The resulting system requirements together with the technical solution of the D^3 system are the basis for the architectural design of the AD4 System.

From the point of view of the MDA technology, the principal benefit of having the D^3 system available is that it significantly facilitates the understanding of what the resulting product has to accomplish and how it is expected to work. Having a correct idea about the operational aspects of a software product is a crucial point in the MDA-development. It enables formal specification by means of models, initially leaving aside the technological and engineering details which are irrelevant to the fundamental functionality of the software system. At the same time, knowing the background system guarantees, to a great extent, the correctness of the architectural solution for AD4. In particular, it ensures that the AD4 components and their interfaces are adequately defined.

As a successor of D^3, a number of specific requirements exist with respect to AD4. Namely, it has to be a distributed component based system, reusing existing D^3 components and providing integration with external, pre-selected platforms.

Additionally, the AD4 development has to be model-driven. In order to minimize the risk of applying the new MDA technology and to ensure rapid development of the AD4 system, agile approaches to software development also need to be put in place.

AD4 life cycle

A System Development Life Cycle is the overall process of developing a software system through a multi step process from investigation of initial requirements through analysis, design, implementation and maintenance. Defining an appropriate life cycle for a project helps to achieve predictable results and to coordinate resources. This is an essential success factor for a project such as AD4, developing big and complex systems, involving specialists from different organizations and countries.

Taking into consideration that the AD4 system (i) is based on the D^3 software, which includes a number of components, to be reused, (ii) is developed by a distributed team of experts in the ATM domain and in software development, and (iii) involves exploration of the new MDA technology, a number of constraints to the life cycle are identified.

More precisely, it has to:

- *Support component-based development*
- *Support model-based development*
- *Be iterative*
- *Support collaboration of distributed teams*
- *Support learning from experience.*

A number of established life cycle models have been investigated with respect to their suitability to the AD4 project. Among these are the Component-based software development life cycle model, Spiral Rapid Application development, Also, life cycles derived from the best known agile methods, Agile Modelling, eXtreme Programming, Feature Driven Development and Adaptive Software Development] have been analyzed.

A key feature of the component-based software development life cycle model is the emphasis on reusability during software creation, and the production of components that are meant to be used in future projects.

Challenges

Reusability implies the use of composition techniques during software development, which is achieved by initially selecting reusable components and assembling them or by adapting the software to a point where it is possible to pick out components from a library.

In general, this life cycle model supports the requirement to build the AD4 system based on components and to reuse components from D^3. However, since D^3 was developed within another R&D project, the system components have not been fully packaged as to be easily reused in future similar applications. This implies performing additional activities related to the completion and the preparation of these components for usage in AD4. Moreover, this has to be aligned with the scope and the effort planned for the AD4 project, i.e. it requires decision making for each D^3 component to be included in AD4.

Another problem is that the AD4 system is foreseen to be integrated in different platforms. Additionally, specific security aspects have to be considered and implemented in it. Since both, the platforms and the definition of the security aspects to be addressed in AD4 are a subject to investigate, the life cycle should provide the necessary flexibility with respect to changing requirements. This is not that easy in the component-based life cycle due to the extent of rework related to completing the components.

The agile methodologies provide features like iterativity, team collaboration and learning from experience. These methods are also suitable for developing products which requirements are rather in a process of investigation. However, the agile methods are not quite appropriate when the product development is carried out by teams distributed in several countries, with different types of expertise and different levels of experience with the technologies to be used. Therefore, only some of the most relevant aspects have been selected to be included in the AD4 life cycle.

With respect to the MDA technology, it is important that the life cycle reflects the development of a tool chain, which will be used to implement the software system. The process of tool chain development includes the following steps (1) identifying the Platform Independent Model (PIM) and the Platform Specific Model (PSM) metamodels (concept spaces), (2) creating model repositories and (3) creating model transformation (PIM $\rightarrow$ PSM) and code generation (PSM $\rightarrow$ Platform).

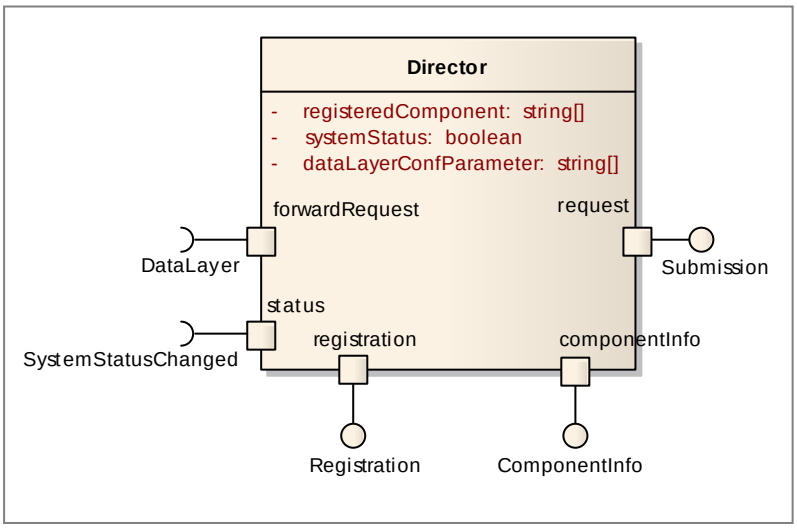


Fig. 1 The PIM of the Director

Based on the considerations about the life cycles related to the AD4 one, two alternative life cycles have been defined and analysed according to specified criteria. The one which has been selected for the AD4 project is illustrated and explained in the next section.

Phases of the AD4 development life cycle

The overall AD4 life cycle is organized in phases (iterations) with each phase being characterized by a series of goals and activities to be performed in order to achieve these goals. Within each of the iterations a small subset of requirements is selected to be developed. At the beginning of an iteration the system requirements are revised: existing requirements can be updated (on the basis of the previous iteration review) and new ones added.

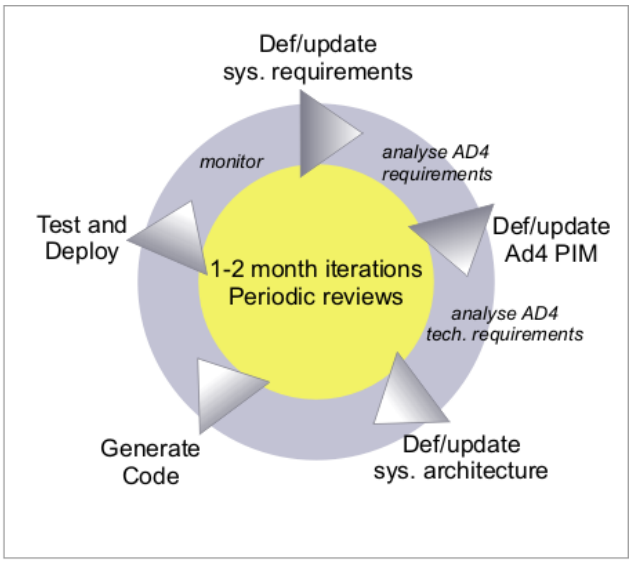


Fig. 2 Cyclic phases in the AD4 life cycle

Changes are clearly propagated to subsequent activities such as platform independent modelling, architectural design, implementation, test and deployment. Each iteration

terminates with a review and a retrospective of all the activities performed within it with the objective to assess development results achieved in the period.

As a result each development phase allows us to incorporate lessons learned into the next iteration. In the following sections we outline the content for each phase and describe what we hope to achieve:

Phase 0: Preparation

This phase aims to prepare the "environment" for later phases and lay the foundations for all of the iterations throughout the project's duration. Key activities of this phase are centered on requirements gathering and planning of infrastructural concerns such as the modelling and development infrastructure (AD4 Tool Chain) and the actual physical and logical platform (preliminary architectural design) on which the system will be built. Later, a subset of requirements for next phase, are selected and a preliminary PIM model is produced.

Phase 1: First release

The aim is to provide the first release of system infrastructure and core components. The development focuses on the implementation of requirements selected in the previous phase. Key activities concern the identification and enhancement of D^3 components to be reused in AD4, the identification of simulation platforms components to interoperate with and the design of integration strategies. System integration and test activities are part of this phase as well. The first PIM to PSM transformation is produced and then, after refining the PSM model, the code for the identified components is generated.

To check the validity of these new components we need to prepare the system integration and test environment. Another key part of this phase is the updating of existing assets and the identification of new components that will need to be developed. In order to include improvements in future phases, a "retrospective" workshop is held, where we decide on any corrective action that needs to be included in the next phases and identify and problematic areas.

Phase 2: Final release

The aim is to provide the final release of system infrastructure and core components. A key is the inclusion of the lessons learned in the previous phase. The activities to be carried out are those already defined in Phase 0 modified according to the conclusions of the retrospective workshop. At the end of this phase we will hold another workshop to provide another feedback loop making sure we continually adapt our process by applying best practices thus mitigating risk throughout the projects development.

Phase 3: Demonstrator

This phase aims to define a scenario, to construct the demonstrator and to integrate it with ATC simulation platforms in order to validate the proposed airspace/airdrome 3D representation. In this phase we will include the components developed up to now and define the test cases for the demonstrator. The lessons learned from this phase will be used for future development work and the whole life cycle will be stabilized.

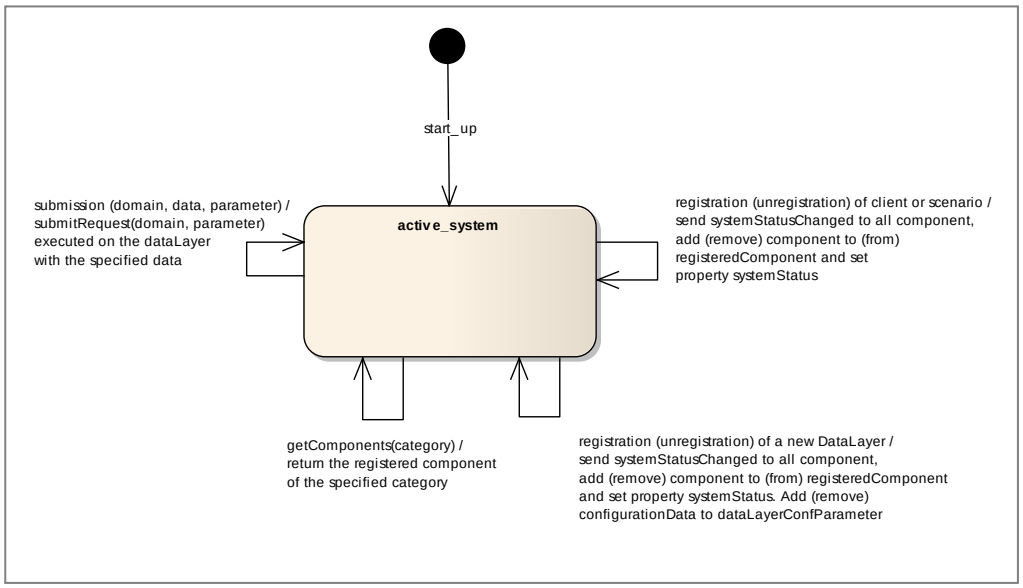


Fig. 3 State chart of Director

Technology Choice

The model-driven development process in the AD4 project consists of two parts: first, designing and building of the AD4 tool chain; second, designing and building the AD4 system by using the AD4 tool chain following the AD4 development lifecycle. The idea is to integrate the most suitable existing development tools for AD4 in one open integrated environment, and implement just model transformers and profiles to make it work. The artefacts that are needed for the tool chain construction are shown in Fig. 4 and explained in the following sections.

Platforms

In order to automate mappings between application models we have to identify and define an executable technology or platform for AD4 system. Indeed, the input to and output from AD4 tool chain steps is directly dependent on the way platforms are used and described.

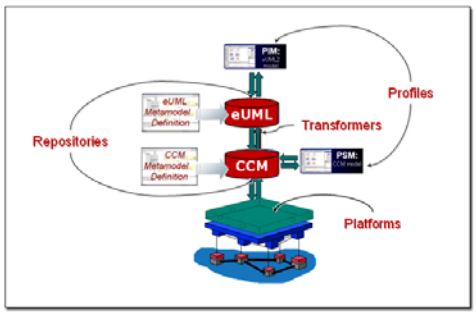


Fig. 4 AD4 Tool Chain building artefacts

The CORBA Component Model (CCM) [1] defines a component model based on CORBA which supports interactions of complex distributed objects written in different languages for different operating systems. All components of the AD4 system are developed as CORBA Components to gain from benefits a component based platform provides with respect to the composition and the deployment of applications. In the AD4 project we use Qedo [2] which is an open source implementation of CCM. Qedo provides code generators, development support, runtime environment, and deployment infrastructure for CORBA Components.

Metamodels and Repositories

Metamodels play an important role in the AD4 tool chain building process because they provide a means to manage models. Out of metamodels we can create repositories where models are stored and managed. For the AD4 project we need both a PIM and a PSM metamodel. For PIM modeling (see Fig 5) we use UML2. However, in order to avoid huge repository size, premature design and to facilitate the comprehensibility of the modelling techniques to the involved user with varying UML2 backgrounds, a specialized subset of UML2 language, namely eUML (essential UML) has been tailored. eUML includes only required UML2 metamodel elements which formally define modelling elements, their semantic and relations. The PSM metamodel is the standardized CCM metamodel as defined by the OMG.

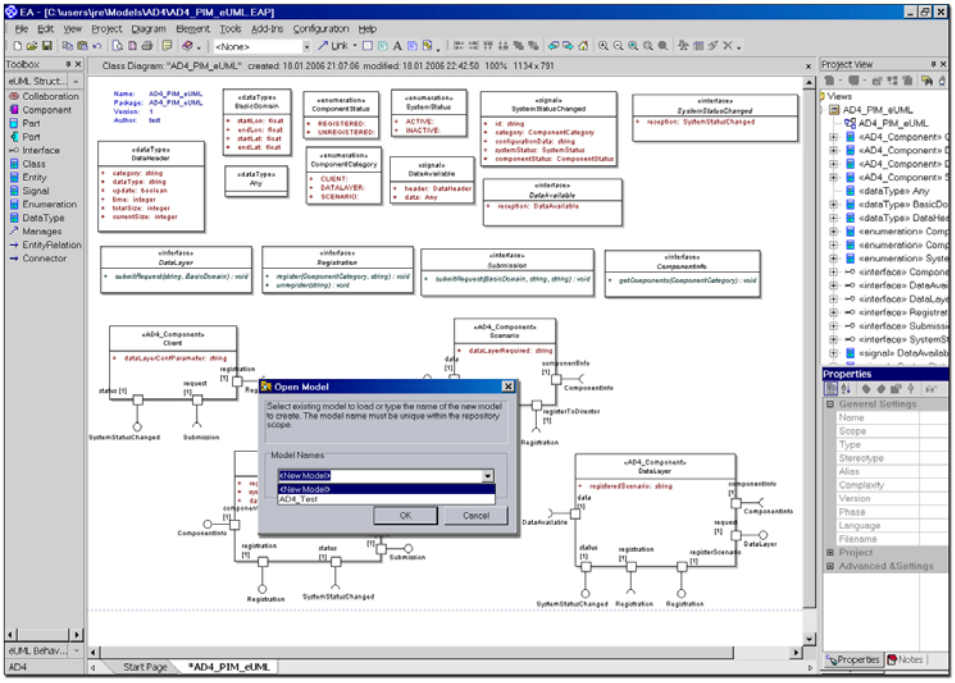


Fig 5: AD4 PIM Model with eUML

Tools and Profiles

Selecting the right tools is essential for building an effective tool chain. Since we use UML2 for metamodel specification and for system design we selected Enterprise Architect (EA) from Sparx Systems as a host UML modelling tool and realised the eUMLModeller, (see Fig 6) which is a Plug-In implementation of the eUML Profile for EA and used for PIM modelling. eUMLModeller is synchronized with the AD4 repository in both directions (load and store of eUML models), including the synchronization of graphical information of the models stored in the repository.

As a modelling front-end for CCM an Eclipse Plug-In, the CCM Modeller, has been developed based on EMF and GEF. The Eclipse Plug-In is a profile implementation for CCM. The Plug-In is also synchronized with the AD4 repository in both directions.

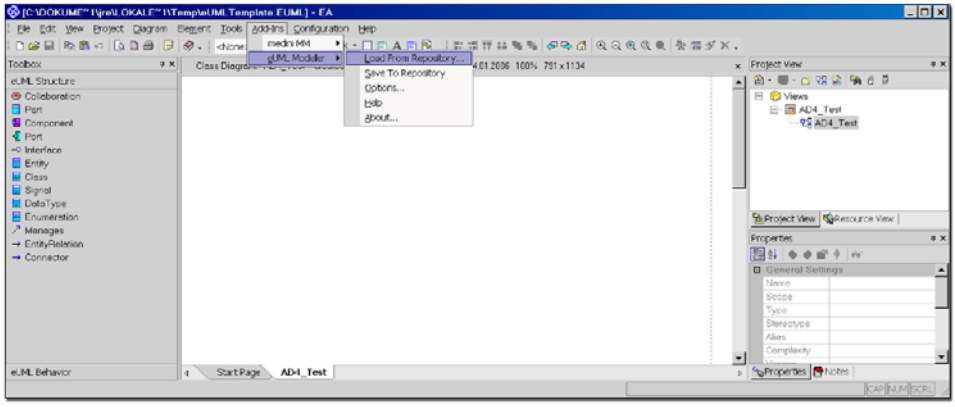


Figure 6: eUML Modeller Plug-In: Connection to the eUML repository

Transformers

To achieve the integration of different modelling techniques and for different modelling layers, the repositories (eUML and CCM) are interconnected together by specific model transformers, which map models to other models or to a programming language code.

Two transformers have been built and integrated into the tool chain: eUML2CCM and CCM2IFR. eUML2CCM transforms eUML models into CCM models. Since Qedo's code generators are based on the Interface Repository (IFR), the CCM2IFR transformer has been developed, which integrates Qedo into the AD4 tool chain. CCM2IFR implicitly generates C++ code for CCM components by triggering the Qedo code generators after transforming the CCM model into an IFR Model.

AD4 Tool Chain Architecture

The heart of the AD4 tool chain (see Fig 7) is a generic control application component, implemented for AD4 and used to manage and control the various components of the tool chain. The AD4 Control Application completely manages the loading of repositories, transformers and models.

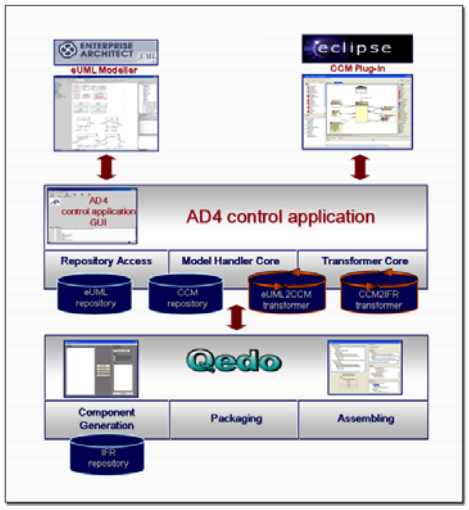


Fig. 7 AD4 Tool Chain architecture

In the standard configuration of the Control Application loads the eUML and the CCM repositories, and two transformers: the eUML2CCM and CCM2IFR.

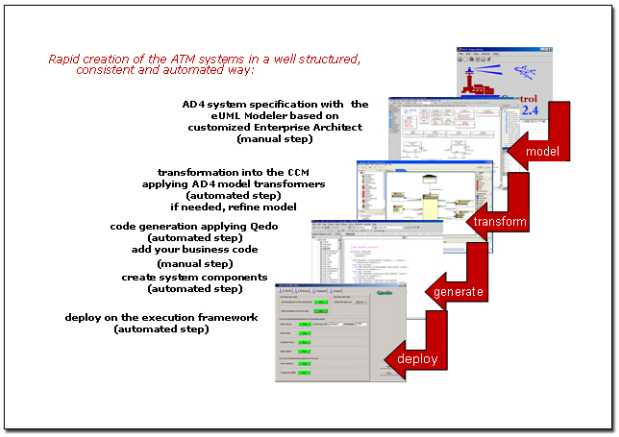


Fig. 8: Applying the AD4 tool chain to system development.

The development of the new AD4 platform starts with requirements specification and analysis. Afterwards, it has to be decided how the existing D^3 platform will be modified, in particular what components will be created or updated and what technologies and techniques will be used for this. In the next step new components are designed using the eUMLModeller. The created eUML model is then ready to be transformed into the CCM model applying the eUML2CCM transformer. The transformed CCM models can be either refined, by using the CCM Modeller, or directly transformed into the Qedo IFR repository which implicitly invokes the Qedo code generators and produces the source code of the CCM components.

Tying it all together

MDA is a new technology allowing the organizations to benefit from the model-centric approach of software development. However, applying MDA requires performing specific practices related to preparing and handling the needed modelling environment, as well as to developing software based on models. These requirements imply specific adjustments to the life cycles which would be used in code-centric approaches to development.

In the AD4 project, the preparation of the modelling environment started together with the investigation of the system requirements and the development of the operational al scenarios, in Phase 0. Normally, the requirements should be traceable to the PIM and PSM constructs that implement them, and vice versa. However, requirements specification in the case of the AD4 project involves considerations of human, psychological and 4D Human Machine Interaction factors, which substantially increases the complexity of this task. Moreover, defining the requirements in a format, which allows integrating them in the tool chain, proved to be an issue that still requires deep investigation. Therefore, the AD4 tool chain begins with a PIM, developed by means of the eUMLModeller, and the system requirements are modelled independently from the AD4 tool chain.

Designing and building the AD4 system by using the AD4 tool chain is iterated through Phases 1 and 2. The approach is to prioritise the requirements for the AD4 system and to start with modelling and creating new or updating D^3 components addressed by the highest priority requirements and proceed like this with requirements having lower priority. The PIM and PSM models are continually synchronized.

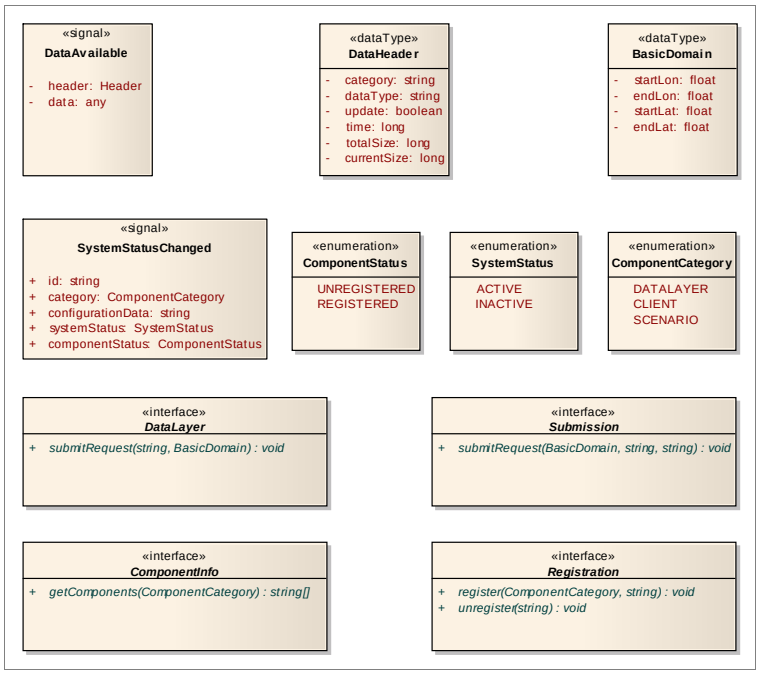


Figure 9a: A PIM of AD4 interfaces, signals, data types and enumerations

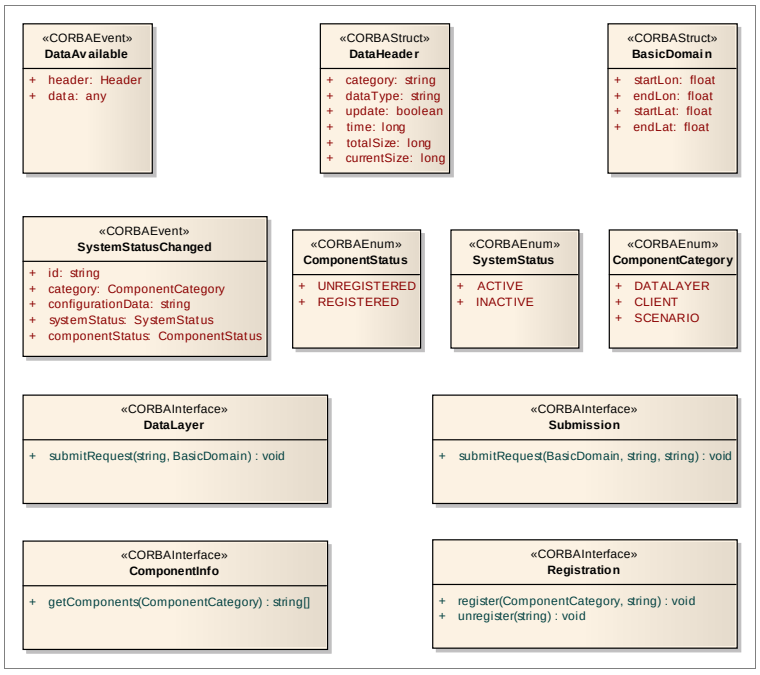
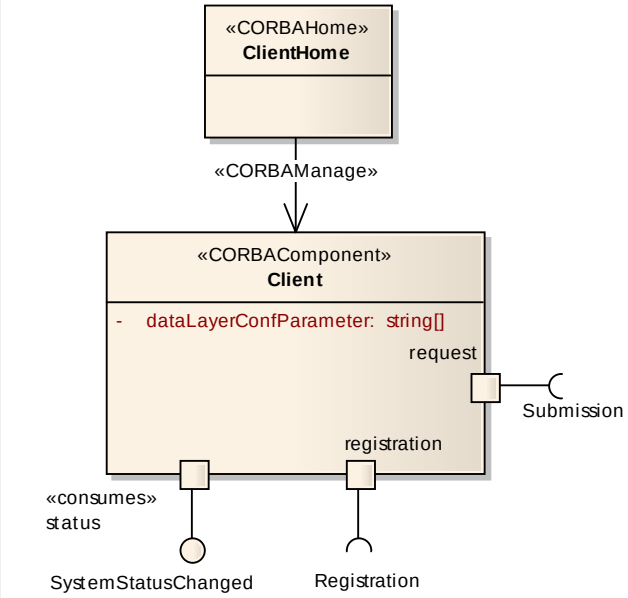


Figure 9b: A PSM of AD4 interfaces, signals, data types and enumerations

At the end of each iteration a key activity is the retrospective workshop, which gathers together all involved in the iteration, to discuss which practices were particularly useful for the joint work and how to improve other practices as to achieve better development results. This activity fosters the adoption of the new MDA technology and the learning from experience.

The proposed tool chain provides the developers with the opportunity to decide later on which platform to run the product. This opportunity is realized by relevant transformers that should be integrated into the tool chain once the target platform is selected. Additionally, since other specific technologies or platforms (e.g. J2EE) can be supported and the eUML models can be transformed to new platform specific models, the list of loadable tool chain components can be arbitrarily extended.




```
component Client {
    attribute string[] dataLayerConfParameter;
    uses Registration registration;
    uses Submission request;
    consumes SystemStatusChanged status;
  };
```

Figure 9c: The PSM of Client & Generated Code

Enterprise Architect

We looked for a tool which had the potential of great acceptance in different domains and companies. Enterprise Architect was chosen for two reasons. First of all, it has a good cost structure and a good cost/value balance with the range of supported UML2 features. Secondly, EA provides a good plugin mechanism which allows the easy integration of custom made plugins. This was vital for the seamless integration within the AD4 tool chain. Finally most important in our consideration for using Enterprise Architect was the extensibility, the cost-benefit ratio and the industrial strength which was the most visible property of EA at this point in time.

It should also be noted that EA also provides a good “out-of-the-box” backbone for MDA adoption and provides the basis on which we could easily extend the solution; especially in the domain of requirements modelling and traceability.

Findings

Our experience in the AD4 project shows three main conclusions:

A practical approach to adopting MDA, especially in complex software development projects is to apply the new technology to extend an existing application. This provides a clear idea about the expected inputs and outputs from the modelling activities, as well as benefiting from domain expertise in the area of development.

Defining a life cycle, which is based on short iterations, active feedback between the designed system and the actual output, and retrospective workshops at the end of each iteration, facilitated the adoption of the new technology and learning from experience.

Developing an adequate tool chain not only speeds up the development process and guarantees the quality of the results, but also increases the flexibility with respect to future developments. For the time being, these conclusions are only based on qualitative data since quantitative ones from the previous and the current project are not yet available.

Using MDA provides a number of benefits to projects like AD4:

- Easier implementation on different platforms while conforming to the system structure and behaviour as described in the PIM.
- Integration of different applications by explicitly relating their models, enabling integration, interoperability and incremental system evolution.
- Easier validation of models uncluttered by platform-specific semantics, since the PIM does not include unnecessary information.
- Clear separation of concerns throughout the development process.
- Improved system quality due to better and earlier fault detection.

Integration of requirements specification within the tool chain remains an open issue and this is the focus of our future work.

References

1. Object Management Group. CORBA Component Model. OMG document number formal/02-06-65

2. Qedo Team. Qedo (Quality of Service Enabled Distributed Objects) CCM Implementation Web Page, http://www.qedo.org/, March 2006

About the authors

Terry Bailey is an R&D Projects Area Project Leader at ESI (European Software Institute). He joined ESI at the beginning of 2005. Until then, and since 1992, he has worked in industry in the fields of Systems Engineering and Software Engineering, working for companies such as Caudwell Communications, Safeway and Thales. His work on Software Engineering has been cantered on the management and development of software applications, mainly using new technologies.

At ESI Mr. Bailey has actively participated, among other projects, in the MODELPLEX IST project as a member of the Technical Coordination Team, Exploitation Manager and member of the Executive Board. He is also the organizer of the successful workshop series "From code centric to model centric software engineering." Mr. Bailey obtained his BSc (Hons) from Staffordshire University and his MSc (Euro) from the University of Brighton/IUT de Bayonne (France).

Tom Ritter graduated with a Masters degree in Computer Science from the Technical University of Berlin. Since 1998 he worked at Fraunhofer Institute FOKUS in the area of tool development and distributed systems. His major interest is the development of model based software engineering tools and the development of component-oriented Middleware platforms with consideration of extra-functional properties.

In his recent work he developed a CORBA Component based Middleware Platform (Qedo) and participated to the design and implementation of the ModelBus as part of the MDDi project. Tom is involved in different standardization activities at the Object Management Group and has contributed to workshops and conferences.

Company Profiles

European Software Institute (www.esi.es)

ESI is a member-based technology research centre located in Bilbao (Spain). ESI has an international team of more than 100 staff members, with extensive technical and managerial experience from research and industry. ESI was created in 1993 by the initiative of the European Commission and with the support of leading European companies and the Basque Government. Since 2003, ESI is integrated in TECNALIA Technology Corporation, which gathers together more than 1,300 highly qualified professionals and researchers working in several technological domains and industrial sectors.

ESI R&D projects are instrumental in developing high-quality technology, products and services that aim to support and promote the adoption of technology in software-intensive industries such as telecommunications, banking, aeronautics, insurance, administration and ICT. Research activities are performed in close collaboration and co-operation with its members and with leading European companies, putting the emphasis on the validation of the approaches by performing experimental trials to ensure the effectiveness.

ESI competencies are on open systems interoperability and standards, model-driven design and reuse through software product-line approach, dynamic reconfiguration and interoperability, COTS integration management, embedded systems development processes and tools, integrated quality, quality of service, certification of products and processes, built-in security, risk and vulnerability analysis and trustability models.

Technology transfer is an essential complement to research at ESI, ensuring the uptake of research results and an impact in industry competitiveness. ESI dedicates much effort to practical dissemination and implementation of technology through its own consultancy and training units and through ESI's Commercial Network, named ESI@net., with partners in more than 50 countries achieving a real multiplication effect and world-wide impact.

The ESICenter network is integrated by a series of independent institutions, which are similar to ESI in their goals, objectives, activities and legal status. Each centre is focussed in supporting the IT industry in a certain region. The ESICenter network complements ESI's existing technological capabilities enabling to launch initiatives at a global level. Currently the ESICenters network comprises seven centres as listed below:

- *ESICenter UNISINOS, Río Grande do Sul, Brazil;*
- *ESICenter SSEAC, Shanghai, China;*
- *ESICenter Bulgaria, Sofia, Bulgaria;*
- *ESICenter Australia, Melbourne, Australia;*
- *ESICenter Tec de Monterrey, Guadalajara, Mexico*
- *ESICenter SECC in Cairo, Egypt*
- *ESICenter Cono Sur, Buenos Aires, Argentina*
- *ESICenter Sinertic Andino, Bogota, Colombia*

Fraunhofer FOKUS (www.fokus.fraunhofer.de)

Fraunhofer Fokus, the Institute for Open Communication Systems is situated in Berlin of the Fraunhofer-Gesellschaft zur Förderung der angewandten Forschung e.V. (FhG). FhG is a link between science and industry, that is between the research and the application of its results. It was founded in Munich in 1949 as a non-profit registered association. The organization takes its name from Joseph von Fraunhofer (1787-1826), the successful Munich researcher, inventor and entrepreneur.

The Fraunhofer-Gesellschaft is an autonomous organization with a decentralized organizational structure, which currently maintains 58 research institutes and a patent office in locations throughout Germany. Whilst the administrative headquarters are in Munich, the legally non independent research institutes operate from different locations in 15 of the German Länder, where they carry out their respective work in close partnership with industry. A staff of approximately 13.000, the majority of whom are qualified scientists and engineers, work with an annual research budget of about one billion Euro.

FOKUS develops a seamless communication infrastructure upon which globally distributed, environment-aware application components can interact according to users' demands. The work of FOKUS covers design, specification, implementation and consulting in the following key areas: model driven development and testing of distributed communication systems, distributed object technology, platforms and services, global heterogeneous networking and internet technologies. FOKUS has gained considerable experience in these areas, for example in EURESCOM projects, TINA auxiliary projects, RACE/ACTS/IST projects, as well as in joint projects with industry partners.

FOKUS has years of experience in the design, specification, and development of open distributed systems. Currently FOKUS is very much engaged in the emerging arena of Model Driven Engineering (MDE) of distributed systems. A new MDE business unit was founded in the beginning of 2003. FOKUS has announced the development of MDA (Model Driven Architecture) related modelling front-ends, code generators and target middleware infrastructures.

MDA and the related supporting software components from FOKUS will be applied in several national and international projects with a telecommunication background and adapted to new application domains like e-government, public safety or air traffic management systems. Together with its industry partner NTT Data Corporation (Japan) FOKUS won the 2003 OMG's Object Application Award for application of MDA concepts in the egovernment domain. FOKUS contributes significantly to OMG standards in the MDE area and subsequently publishes papers for scientific conferences and workshops.

About Sparx Systems

Sparx Systems *(www.sparxsystems.com)* specializes in high performance and scalable visual modeling tools for planning, designing and constructing software intensive systems.

With customers in industries ranging from aerospace and automotive engineering to finance, defense, government, entertainment and telecommunications, Sparx Systems is a leading vendor of innovative solutions based on the Unified Modeling Language (UML) and its related specifications. A Contributing Member of the Object Management Group (OMG), Sparx Systems is committed to realizing the potential of model-driven development based on open standards.

The company's flagship product, Enterprise Architect, has received numerous accolades since its commercial release in August, 2000. Now at version 7.1, Enterprise Architect is the design tool of choice for over 150,000 registered users in more than 60 countries world wide.

Project Profile (Other partners & consortium overview)

The AD4 Consortium included research as well as industrial partners committed to the acquisition and development of the advanced technological know-how required to address the project objectives and to the exploitation of the results achieved through a specific ATM infrastructure and a definite demonstrator. Final users and testers of the targeted ATM field were also key contributors to the project development and assessment: They produced specifications of real operational needs within their business cases and co-ordinated the final phase of testing and validation of the prototype. Finally, SMEs were well represented in the consortium which was made up of the following companies: NEXT Ingegneria dei Sistemi S.p.A. (Project Co-ordinator), ENAV S.p.A., VITROCISET SpA, MIDDLESEX UNIVERSITY, Space Applications, Digital Video, SICTA (Sistemi Innovativi per il Controllo del Traffico Aereo), Fraunhofer FOKUS, European Software Institute and ObjectSecurity.