	Deliverable reference: D.8.9	Date: 21.05.2007
	Title: **Final Report**	
Project Title: Mobility and ADaptation enAbling Middleware **MADAM** Proposal/Contract no.: 4159	Responsible partner: SINTEF	
	Editor(s): Sebastiano Lombardo	
	Approved by: Geir Horn, Project Manager	
SIXTH FRAMEWORK PROGRAMME: PRIORITY 2.3.2.3 SPECIFIC TARGETED RESEARCH OR INNOVATION PROJECT	Classification: Public	

Abstract:

This document presents an overview of the results achieved by the MADAM project and highlights from the work performed by the consortium.

The challenges behind the project conception are described along with the way the MADAM project tackled them. The results are highlighted presenting scenarios and descriptions of the pilots performed during the project period. The reader will also find useful information on the adaptation theory and its possible applications.

Suggestions for future work and enhancements to the middleware platform and supporting tools build in MADAM is also provided.

Key words:
Project Management, Strategic Challenges, Project Results, Project Impact, Application Pilots.

Table of Contents

1 EXECUTIVE SUMMARY 3
2 THE CHALLENGE 4
3 ADDRESSING THE CHALLENGE: THE MADAM PROPOSITION 5
4 WHO CAN BENEFIT FROM MADAM 6
5 HIGHLIGHTS AND ACHIEVEMENTS 7
6 THE RESULTS 8
6.1 ADAPTATION SCENARIOS 8
6.1.1 The Janitor Scenario 8
6.1.2 The VSAT Installer Scenario (SatMotion) 9
6.1.3 The Student Scenario 9
6.1.4 Reference Requirements 10
6.2 THEORY OF ADAPTATION 11
6.3 METHODOLOGY AND TOOLS 12
6.3.1 MADAM Methodology 12
6.3.2 MADAM Tool Support 13
6.4 ADAPTATION MIDDLEWARE 14
7 THE PILOTS 17
7.1 THE SERVICE TECHNICIAN APPLICATION 17
7.2 THE SATMOTION APPLICATION 17
7.2.1 The original SatMotion System 17
7.2.2 The adaptive version of SatMotion 18
7.2.3 Development process: Adding adaptation to SatMotion 19
8 AVAILABILITY OF THE RESULTS 27
8.1 SCIENTIFIC IMPACT 27
8.2 ECONOMIC IMPACT 27
8.3 IMPACT ON STANDARDS 27
8.4 IMPACT ON POLICY 27
9 LESSONS LEARNED 28
10 THE CONSORTIUM 31
10.1 THE CONSORTIUM 31
10.2 CONTACTS 34

1 Executive summary

Traditionally, software systems have been built to satisfy a fixed set of requirements, often making them unsuitable in settings where user needs, and operating conditions, vary dynamically. With the increasing mobility and pervasiveness of computing and communication technology, the ideas of mobile and ubiquitous computing are becoming a reality. This introduces significant dynamic variation both in the user needs and in the operating environment for the provided services. In this situation we need ways to deal with the construction of self-adapting systems that are affordable also for "everyday" systems.

MADAM's approach to meet these challenges builds on the following main ideas:

- Component frameworks as a way to build both applications and middleware that are capable of being adapted by reconfiguration.
- Model driven development emphasising modelling of adaptation capabilities and context dependencies.
- Implementation of context monitoring and adaptation reasoning and reconfiguration as generic algorithms, leveraging the resulting models at runtime.

MADAM offers to developers powerful features to add adaptation to applications. We emphasise the concepts and notations for modelling of adaptation capabilities,and context dependencies, and their theoretical foundation. Combining this with technologies based on the Model Driven Architecture (MDA) philosophy has enabled a comprehensive tool support for our approach.MDA is an emerging technology and the achieved results have been very encouraging.

The technology developed in this project has the potential to enable a broader set of ubiquitously available services both for business and private use. The adaptation of services will improve the service usability and other service quality properties in mobile settings. As a result the project may have a positive impact on the everyday life of citizens who use public and commercial services, the service providers (public and commercial), application developers, communication infrastructure operators, handheld device manufacturers and vendors along with the research community. Concrete examples are provided using three scenarios ("the janitor", "the satellite antenna installer" and "the student"). Two pilot applications supporting aspects of these scenarios have been implemented to demonstrate and assess the MADAM technology.

In conclusion, the MADAM project has delivered several innovative solutions for self-adaptive applications. Furthermore, it has opened up a set of new technical challenges that deserve further research.

2 The Challenge

Traditionally, software systems have been built to satisfy a fixed set of requirements, often making them unsuitable in settings where user needs, or operating conditions or both, vary dynamically. In some domains where continued utility have to be ensured despite dynamic variation in the operating environment, more flexible systems have been built, for instance software for controlling telephone exchanges or space vehicles. However, such flexibility has generally been very costly to implement.

With the increasing mobility and pervasiveness of computing and communication technology, the ideas of mobile and ubiquitous computing are becoming a reality. More and more software systems are used on, or accessed by, a variety of handheld networked devices used by people moving around. This introduces significant dynamic variation both in the user needs and in the operating environment for the provided services. For example, communication bandwidth change dynamically in wireless communication networks and power is a scarce resource on battery powered devices when outlet power is not available. Furthermore, user interface preferences change when on the move, because light and noise conditions change, or because hands and eyes occasionally are busy with other things. Dynamic adaptation is required in order to retain usability, usefulness, and reliability of the application under such circumstances.

In this situation we need ways to deal with the construction of self-adapting systems that are affordable also for “everyday” systems.

Self-adapting systems that have been built so far have typically been mission critical systems in some sense, where software development cost has not been a limiting factor and considerable additional development effort to achieve the necessary flexibility has been acceptable. For self-adaptation to be feasible in “everyday” systems, more cost effective ways to achieve self-adaptation must be found that make it affordable also for this class of systems. We believe that an approach to achieve this must satisfy the following requirements:

- ***Provide benefit**:* Clearly there must be a perceived benefit in the sense that self-adapting applications are better able to maintain its service level and satisfy user requirements during variations both in operating conditions and user requirements. However this does not mean that adaptation must always be perfect in the sense that all user requirements must be fully satisfied at all times. Rather we are talking about providing sufficient utility for everyday users, much in the same way as discussed by Shaw[1].
- ***Limit runtime overhead**:* We are targeting mobile use, which means that terminals often are small and have limited resources. Also the communication capacity will often be limited on wireless links. It is important that the adaptation middleware does not impose intolerable overheads with respect to these resources
- ***Limit developer burdens**:* To ensure acceptance by broad developer groups, the approach must align well with commonplace development methods, languages and tools, and limit additional burdens on the developer.
- ***Component based**:* Everyday does not mean small and simple. Most everyday systems are large and complex enough to necessitate a modular development approach. Thus developing self-adaptation capabilities must also be possible on a per-component basis, and must not compromise the possibility to compose applications and the use of 3rd party components.

[1] Shaw, M. "Everyday Dependability for Everyday Needs". Keynote speech at ISSRE 2002

- ***Exploit reuse**:* Adaptation capability is a common concern for a large class of systems and it would be a long step towards affordability if it could be supported, to a large extent, by generic reusable components.
- ***Separate concerns**:* Even though many tasks related to self-adaptation may be handled by generic reusable components, achieving self-adaptation also require application specific development. It is important that this development, and its results, are separated as clearly as possible from the normal development in order to keep complexity under control

3 Addressing the challenge: The MADAM proposition

In MADAM we address these challenges by an approach that builds on the following main ideas:

- Component frameworks as means to build both applications and middleware that are capable of being adapted by reconfiguration.
- Model driven development based on extended UML to capture adaptation capabilities and context dependencies in architectural models.
- Representation of such enriched architectural models at runtime as way to enable generic algorithms for making adaptations.
- Implementation of context monitoring and adaptation management as generic middleware.

The MADAM approach is illustrated in Figure 1. The middleware manages a set of possibly distributed applications running on a distributed computing environment consisting of a networked handheld device and possibly a number of other devices in its immediate surroundings, as defined by the wireless network. At the heart of the middleware is a control loop with the following main steps:

- ***Detect context changes***. Context consists of elements representing the computing infrastructure (such as battery level and network resources) and the user environment (such as position and background noise) and the user needs.
- ***Reason about the changes and make decisions about what adaptation to perform***. When context changes occur, we need to re-design the system, i.e. find a new design that better fits the new context and thus may re-establish satisfaction of user needs.
- ***Implement the adaptation choices.*** When a new design has been selected, the middleware re-implements the system in accordance with this new design.

The development of self-adapting applications is supported by a UML profile that allows the insertion of variation points in the architecture and annotations of the model that describes how the resolution of variation points influences the properties of the application, both related to function, offered QoS, and needs for computing and communication resources. Any UML tool that supports profiling can be used. A model to text transformation tool is provided to support the transformation of the of the UML model to the form usable by the middleware.

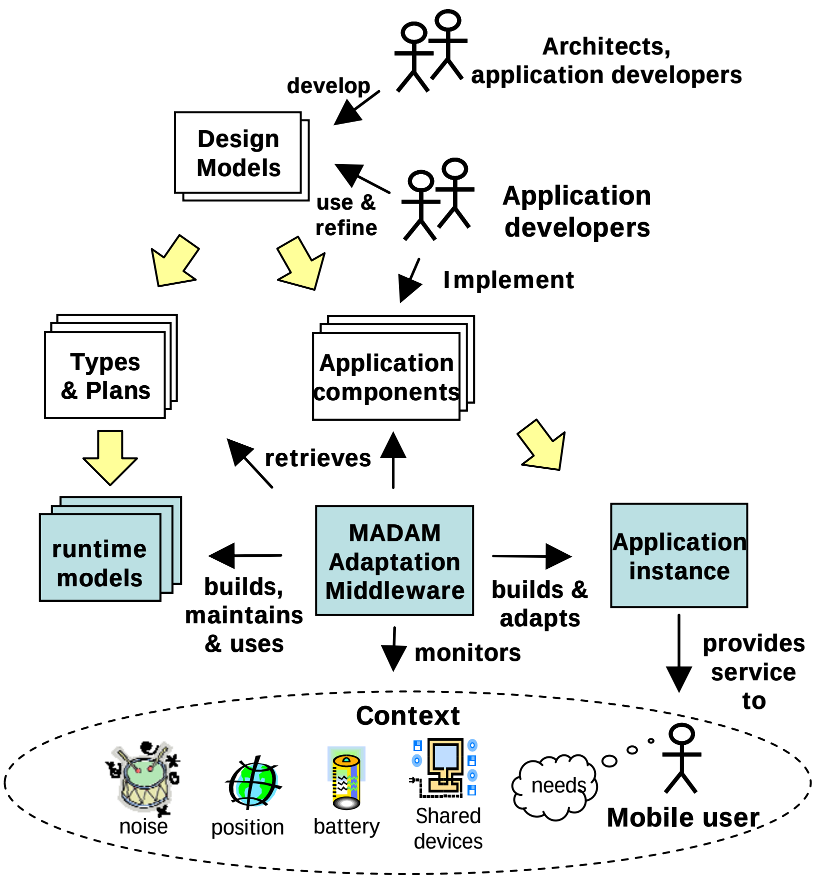

Figure 1: MADAM approach

4 Who can benefit from MADAM

The results of the MADAM project may have a positive impact on the everyday life of several groups of actors. For example citizens who use public and commercial services, the service providers (public and commercial), application developers, communication infrastructure operators, handheld device manufacturers and vendors along with the research community. Next chapter summarizes how the technological achievements of this project concretely affect these actors. Furthermore a few scenarios are presented in this report (see chapter 5.1) to give concrete examples.

The first scenario is about maintenance workers who use both mobile and infrastructure-based computing devices in order to receive assistance from specialists with their everyday tasks. Janitors, who are the main entities of the scenario, are assumed to be skilled technicians who are assigned to several scheduled tasks according to both the regular and acute needs of their customers. The janitors can communicate with the specialists in order to download or upload data sheets, or to get direct guidance and advice.

Another scenario describes the activities of a satellite antenna installer, who uses handheld devices and the SatMotion application, which provides him with assistance during the antenna line-up, and the deployment of a number of installations throughout a day.

Then there is a student scenario based on both realistic situations, and fictive ones as of the time of writing. This scenario describes the activities of a student who attends classes and does research while using different electronic devices. Through this scenario, several adaptations are studied related to time, network connectivity, user’s ability to interact with the device, *etc*.

A number of other user cases can be imagined. For example car driver who, while driving their car, is offered the option of connecting a mobile device to an installed on-board computer, in order to enable a hands-free mode of interaction with their mobile device.

User cases can also be focused on user and application session redeployment. In some changing contexts, the adaptation implementation involves the redeployment of components, or applications on different devices. This can facilitate better and more efficient use of the available infrastructure. For example at the office, the user session can be redeployed from a handheld device to a desktop PC, which offers significantly better user interface capabilities. When leaving the office, the redeployment on the opposite direction can guarantee the continuation in the user's activities.

Finally, the research community itself benefits from the achievements of the MADAM project. The theory on adaptation developed in the project, with a focus on adaptation strategies and decision making will have a potential impact in the research community. The work on reflective, dynamic reconfigurable component architectures may draw similar interest. In universities and research institutes the open source versions of the generic parts of the adaptation middleware is likely to spawn further research activity. We also believe that the work on modelling language extensions for adaptation and quality attributes will have an impact in the Model Driven Architecture (MDA) research community.

5 Highlights and achievements

The technology developed in this project has the potential to result in a broader set of ubiquitously available services both for business and private use. The adaptation of services will improve the service usability and other service quality properties in mobile settings. Also, the handling of quality attributes of the application will lead to more dependable solutions, and will improve the trust in the applications. Overall, this will enable increased use of computer based services in mobile settings, giving service users better support for their tasks when they are moving around.

The last years of evolution of mobile applications and services showed that the most important factor for success and acceptance by users and customers is *usability*. There is a high expectation to have the same comfort of usage available on mobile devices as on traditional desktop-based applications. Many of the mobile applications available today do not meet these requirements, and this leads to a low level of acceptance of publicly available mobile services. The project has focused on a generic approach to solve these issues and deliver services having dynamic adaptation mechanisms. This will make it possible for service providers to cover a wider range of applications and to reach a larger customer base with their offerings.

It also, potentially, allows them to differentiate multiple levels of service quality in relation to which adaptation mechanisms can be provided, and to structure their products accordingly. Altogether the application of adaptation mechanisms developed in this project strengthens the mobile application market and provide arguments for network operators for using UMTS and its capabilities. Service providers will be enabled to distribute their services through different infrastructures.

Moreover the project contributed to the evolution of methodology into software tools. This will ease the process of developing mobile applications with dynamic adaptation in mind. This will also increase the overall quality of mobile services from the user's point of view, and will reduce the effort and therewith the risk commercial application developers take when investing in new products. Another interesting aspect is that device adaptation mechanisms will ensure that services and applications will not become obsolete with the distribution of new mobile devices and standards. This is very important because today the renewal cycles of mobile devices are less than one year, while the lifecycle of applications in the B2B area are between 3 and 4 years.

From the point of view of communication infrastructure operators it can be argued that the project's results contribute to an increased use of the communication infrastructure owing to an increased use of mobile services. The operators of the infrastructure will see an increase in the demand for bandwidth as well as higher frequency of use. Increased availability and use of adaptive mobile services will also increase the demand for handheld devices capable of running these services. Device manufacturers providing good support for adaptation on their platform will have a competitive edge over other manufacturers, and will benefit from adopting the MADAM concepts and middleware.

6 The Results

6.1 Adaptation Scenarios

A set of adaptation scenarios was specified in order to better facilitate the common understanding among the partners, and to determine the requirements for the targeted middleware early in the project. These scenarios were designed so that they illustrate the need for different adaptations, and secondly, to study and assess the existing state-of-the-art in adaptable software systems. The three resulting scenarios were based on suggestions and contributions from all project partners, and they involved both fictional and realistic situations, with some of them based on existing products, developed by some of the project's industrial partners.

Through a detailed elaboration of the scenarios, the consortium has also converged to a set of relevant adaptations. Generally, these adaptations have the form of a *context change* and the relevant, resulting *adaptation*, thus forming a set of requirements in these two areas.

The following paragraphs briefly discuss the three main scenarios that were designed for directing the MADAM effort throughout the design and development of the middleware and the supporting tools. Although these scenarios primarily concentrate on the interactions between the computing infrastructure and the end-user, the reader is reminded that one of the primary aims of MADAM was to offer developers with the appropriate design and development tools. Although these scenarios do not explicitly discuss the development process, it is nevertheless understood that the scenarios indirectly contribute to the understanding of the MADAM developers, by providing them with intuition as to what kind of adaptation services the end-users are expecting.

6.1.1 The Janitor Scenario

The first scenario is about maintenance workers who use both mobile and infrastructure-based computing devices in order to receive assistance with their everyday tasks. Janitors, who are the main entities of the scenario, are assumed to be skilled technicians who are assigned to several scheduled tasks according to regular and acute needs of their customers. Janitors are responsible for the inspection and maintenance of technical equipment such as air conditioning equipment, fire alarm installations, *etc*. Typical customers are large organisations possessing buildings with technical installations spread across multiple and geographically distributed sites. In order to perform their work, janitors use networked handheld devices. These devices provide access to services and applications that facilitate the inspection and maintenance work. In this scenario it is assumed that all the equipment is electronically registered and information about them is available online through the network. Furthermore, the equipment uses *Integrated Circuit* (IC) Technology systems which facilitate automatic tagging for identification, or for the description of entities, e.g.. through WLAN or Bluetooth. By using this technology, some equipment can also be managed and controlled through software applications. In addition to inspection services, handheld devices also facilitate administrative tasks and ordinary communication. This is done to assist with the retrieval of scheduled tasks, and to allow the janitors to communicate with their base in order to retrieve driving directions, technical assistance, *etc*. Specialists are highly-skilled technicians that are located at some unspecified location, and who are available by telephone or other electronic means. The specialists can provide advanced information with regards to equipment adjustments, based on their specific expertise. The janitors can communicate with the specialists in order to download or upload data sheets, or to get direct guidance and advice, if for example they are inspecting equipment they are not familiar with. The exchanged data, which may be textual data, images, presentations or even real-time video, is expected to be presented in a suitable form on the janitor's device.

Throughout a detailed elaboration of the Janitor activities, the consortium has concluded a set of relevant adaptations. These include adaptations such as choosing to download large email attachments when the handheld device is connected to a cheap and efficient network, or choosing to adjust the user interface characteristics like display brightness and speaker volume, according to the ambient light and noise conditions, and even switch between hands-free and visual interaction modes depending on the user's occupation and ability to visually interact with the device.

6.1.2 The VSAT Installer Scenario (SatMotion)

This scenario was originally proposed by the partner INTEGRASYS and involves an existing application which is in commercial use. The scenario was further enriched with input from additional contributions by other, primarily industrial, partners. This scenario describes the activities of a satellite antenna installer who uses handheld devices and the SatMotion application to assist him with the deployment of a number of installations throughout the day. In order to facilitate his tasks, the installer uses the SatMotion application, which provides him with assistance in the antenna line-up. The SatMotion implements the client side of an extensive, distributed application, which provides assistance for installing VSAT antennas. This is achieved by displaying real time remote monitoring data of the antenna line-up, typically a *Continuous Wave* (CW) signal, in the handheld device be it a Smart-phone, a PDA, or a laptop. The communication is carried either through a GPRS or a WiFi connection. The installer starts the line-up procedure by transmitting a test CW signal and adjusting its CW Effective Isotropic Radiated Power (EIRP) while at the same time obtaining an immediate feedback of the action on the client device. The feedback data is initially received by the spectrum analyzer at the hub station, which is then delegated to the corresponding installer via the Internet protocol and a WiFi or GPRS connection.

The SatMotion application provides three functional modalities: The *Two Way*, which implements a two way communication tool system, and the *One Way*, which is a simplified version allowing one way communication only. The second modality implies that signal information can be communicated only by the client to the server, but not the other way. Finally, a third modality, named *Offline* supports limited operability when there is no network available.

The adaptations in this case include situations where several context changes, such as a *low battery* indication trigger adaptations to the software and switching to a more power-efficient mode, and to the hardware when selecting a more power efficient network adapter, or lowering the display brightness to save on power consumption. Furthermore, based on the availability and cost of networks, the application is expected to be capable of switching between any of the three modes in an automated manner.

6.1.3 The Student Scenario

The student scenario is based on both realistic and fictional situations not yet supported by current technologies. This scenario describes the activities of a student who attends classes and does research while using different electronic devices. Through this scenario, several adaptations are studied related to time, network connectivity, user's ability to interact with the device, *etc*.

The scenario describes typical activities of a college student. For example, the student attends courses, but also works as a researcher in a lab in the campus. During a typical day, the student uses several stationary and mobile devices to assist her into accomplishing her every-day tasks in more efficient and automated way. For example, the student uses a smart-phone to get information about the class schedule, get informed of latest updates concerning the homework, deadlines, *etc*, read her email, and, naturally, place phone calls. The used applications are adapted to context changes like available networks, student's activities, *etc*, so that the overall user experience is improved.

The adaptations in this case are detected by studying and analysing typical activities in the student’s day. For example, the day starts with the student driving her car to the campus. She carries a smart-phone that she uses to organise her personal data, and to manage her agenda. She also uses a mobile application, called Student Assistant. The students use their ID number by which the application can request updated information about their class schedule, available presentations, upcoming seminars, *etc*. The agenda is automatically updated as soon as there is an appropriate wireless link available. Furthermore, the student has an installed car computer which supports Bluetooth connectivity and offers hands-free audio functionality for the handheld device. By pairing with the car computer, the handheld computer is automatically updated with information on the user’s activity, i.e. driving, and adapts its mode of interaction accordingly. For instance, incoming email messages are automatically read using text-to-speech technology while the student is driving. Finally, when the student arrives to the classroom, the day’s lecture documents, presentation and notes are already automatically downloaded to the handheld device, because the system is aware of the student’s agenda and that she will be attending course X at time Y.

6.1.4 Reference Requirements

As a result of the *adaptation scenarios* study, the MADAM partners have concluded to a set of interesting reference requirements. These requirements are expressed in the form of *adaptations*, along with the possible context changes that could trigger them. These adaptations have provided significant input to the developers of the middleware and the development tools, as they provide important insight concerning the type of context changes and resulting adaptations that should be supported. The following describes the complete set of reference requirements, along with a short description explaining each one of them.

- *User interface delegation*: This form of delegation involves the transfer of the user interface functionality partly or fully to another device or peripheral. For example while driving their cars, mobile users are offered the option of connecting their mobile device to the installed computer, in order to enable hands-free mode interaction with their mobile device.
- *User Interface presentation*: This adaptation refers to actions required for the tuning of the corresponding user interface device, so that the experience delivered to the end user is optimized for the running context conditions. For example, based on the ambient light conditions of the physical environment, a mobile device can automatically adjust its display brightness.
- *User and application session redeployment*: In some changing contexts, the adaptation implementation involves the redeployment of components, or applications on different devices. This can facilitate better and more efficient use of the available infrastructure. The redeployment of applications to devices providing different user interface properties could be considered as a special case of this type of adaptations. For example at the office, the user session can be redeployed from a handheld to a desktop PC, which offers significantly better user interface capabilities. When leaving the office, the redeployment in the opposite direction can guarantee the continuation in the user’s activities.
- *Functional richness*: Functional richness refers to the extension of the functionality of an application, by providing access to newly discovered software and hardware components or services. For example, an application which is used to tune the home-cinema experience in a

living room, could extend its adaptation domain if a new hardware plug-in was added, offering the capability of automatically controlling the lights in the room.

- *Data richness*: Data richness refers to the case where the quality of a data stream is affected. For example, in the case where a video-conference is affected by the network characteristics such as the network bandwidth and latency, an adaptation might be required to ensure that the best Quality-of-Service is provided.
- *Network availability*: The variability of mobile network technologies (GPRS, WLAN, Bluetooth, *etc*) imposes the requirement for network adaptation and selection. For instance, when many networks and network technologies are available, the device is offered the possibility of switching between multiple networks like a public, insecure WiFi access point, or a corporate secure network through Bluetooth to a desktop. Additionally, offline mode when no networking is available and online modes can also be offered depending on the network context conditions. There are potentially unlimited mobile scenarios which demonstrate the use of switching the network use. For example, a mobile device can stay connected to a WLAN connection when that is available, benefiting from the high bandwidth and the low cost. On the other hand, when the device moves to an area without WLAN coverage, then switching to a GPRS connection might be the only option for maintaining connectivity.
- *Security*: This type of adaptation refers to an appropriate adjustment of the security mode, to compensate for changes in the device’s context, and especially the networking infrastructure. For example when the mobile user is using a GPRS communication link, an encryption mechanism is enabled to secure the communicated data. When the user arrives to office and connects to the corporate network, this mechanism is disabled as the office network is assumed to be safe so the device saves power, processing capacity, *etc*.
- *Software mode*: Finally, the software mode adaptation refers to the option of switching among different modes of operation, in order to meet the application context requirements. It is assumed that the application developers make a number of modes available, and furthermore, they provide guidelines on how to decide which mode is the most appropriate for each situation. Different modes could correspond to different software architectures, such as for example in a video streaming system which allows the selection of different compression algorithms. From a point of view, this is the most general adaptation type, and all other types could be assumed to be specialisations of this one.

6.2 Theory of Adaptation

The Theory of Adaptation establishes a theoretical foundation for context awareness and self-adaptation. It provides an extended State of the Art that covers adaptation, context-awareness and reconfiguration. Beyond this State of the Art, it defines generic and platform-independent mechanisms for reasoning about adaptation and for detecting and analysing context changes. All engineering results developed in the MADAM project are based on this theoretical foundation.

While MADAM tailors existing solutions for context management and reconfiguration for the special needs of mobile computing and adaptive systems, original solutions have been developed for adaptation management. A close analysis of the problem of adaptation shows that the selection of the “best configuration” in a given context is far from obvious, and adaptation management requires complex reasoning for making decisions about adaptation. Therefore an approach based on using utility functions is proposed. In addition to compositional variability, which is used to specify structural and algorithmic system component variability, the approach also supports fine-grained adaptation using parameterization. Further the coordination of the adaptation of multiple applications and the adaptation of the mobile device are discussed.

In the context of mobile computing, the scarcity of processing resources is a major issue. Two main approaches are considered to address the resource constraints on handheld devices. One seeks to limit needs for storage or processing on a device through optimisation of the adaptation algorithms, the other proposes to share memory and processing loads through distribution. Optimisation and distribution can be exploited at all stages, i.e. for context management, adaptation management and reconfiguration. In addition to the problem of scalability, the proposed approach raises several feasibility issues. The Theory of Adaptation discusses the dynamics of adaptation such as the impact of adaptation on context and on service quality, context instability and convergence of the adaptation algorithm, and occurrence of context changes during adaptation or reconfiguration.

In order to facilitate understanding the concepts and the potential of the approach, the Theory of Adaptation addresses adaptation from the viewpoint of the stakeholder. In the developer's viewpoint, the tasks of the developer of adaptive applications are introduced and a set of developer and user roles involved in the development and use of MADAM-based applications are proposed. The user viewpoint concentrates on the user intent: a main goal is to provide the user with the most adequate environment to perform his tasks. The service provider's viewpoint considers the ability to provide adaptive services and extract requirements focused on the needs of service providers.

Finally, although the focus of MADAM project is mobile computing, the Theory of Adaptation briefly introduces to autonomic computing and adaptation in the setting of autonomic computing, and relates the MADAM approach to that research area.

6.3 Methodology and Tools

In MADAM, one of the main objectives has been the exploitation of the Model Driven Architecture paradigm in building adaptive applications. The adaptability of applications is modelled during the modelling phase in an abstract way. According to the MDA-based development process, the adaptation model is automatically transformed to source code using appropriate tools. Thus, in MADAM we have provided a comprehensive methodology that is a guideline to the developer, while building their adaptive applications.

6.3.1 MADAM Methodology

The MADAM Methodology consists of the following step by step approach:

1. *Analysis of Requirements*:As a first step, the application developer must analyse the requirements of the application. Thus he will find the context and resource dependency of the application based on the possible application scenarios. He will also have to analyse the involved components, their communications and the possible variants of those components. For the MDA-based development, the developer should also have a clear idea about the modelling requirements of the application.
2. *Modelling Adaptibility:* After the requirement phase a model of the application is built. We provide a step by step modelling Approach. In MADAM, the modelling is done in UML 2.0. We have developed a new UML profile that covers all the required MADAM concepts for modelling adaptability of applications.
3. *Model Transformation*: MADAM provides model-to-text transformation support using the MOFScript tool, developed in the MODELWARE project. Thus the UML model can be automatically transformed to Java source code which is used by the middleware at runtime.
4. *Code Completion*: The format of the utility function and property predictors is quite flexible. Therefore, in the model only a skeleton is provided for these functions, while their actual implementation is to be decided by the developer. The developers have to complete the code related to the utility function and property predictors. In the generated code, these places are marked as 'TODO' in order to facilitate the identification of those places where the source code must be enhanced.
5. *Packaging and Deployment*; The generated code can not be directly deployed to the MADAM middleware. It is packaged in a .jar file before deploying to the middleware. The users can then install the application by selecting this .jar file.

6. *Testing and Validation*: After deploying the application, the developer has to validate the application for functional correctness. He has to come up with some possible scenarios and observe the adaptation behavior. In the MADAM middleware, we provide a graphical user interface that can be set to 'Simulation Mode' in order to perform these tests. In addition, there are instructions to test performance metrics as well.

6.3.2 MADAM Tool Support

In MADAM, a comprehensive tool chain has been developed, as presented in Figure 2.

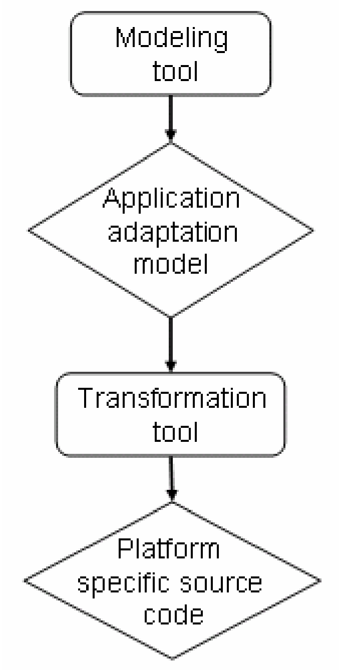

Figure 2: MADAM tool chain

Tool support is provided for the modelling of the application as well as for the transformation of the adaptation model to source code.

- *Modelling Support*: For the modelling of MADAM applications, we have developed a MADAM-specific UML profile extending the UML 2.0 metamodel elements and thus supporting the modelling of adaptation related concepts. Most of today's UML tools support importing profiles and using the stereotypes during the application modelling. In our work, we used Enterprise Architect as the first choice modelling tool.
- *Transformation Support*: The model transformation is based on MOFScript, a model-to-text transformation tool from the MODELWARE project. One of the requirements, however, of MOFScript is that it requires as input a model built on the Eclipse UML2 metamodel (supporting the Eclipse Modelling Framework, EMF), which is a subset of the OMG UML 2.0 metamodel. Only a few modeling tools like Omondo and Borland Together Architect can directly export a UML model to a MOFScript-readable XMI representation. Therefore, for other tools that support modelling in UML 2.0, but do not conform to UML2, a conversion is required. We provide such a conversion for models built on Enterprise Architect. An XSLT stylesheet is available, which can convert the exported XMI corresponding to the UML 2.0 of the Enterprise Architect model to an XMI file corresponding to the Eclipse UML2. The actual realization of the MADAM tool chain of Figure 2 is presented in Figure 3.

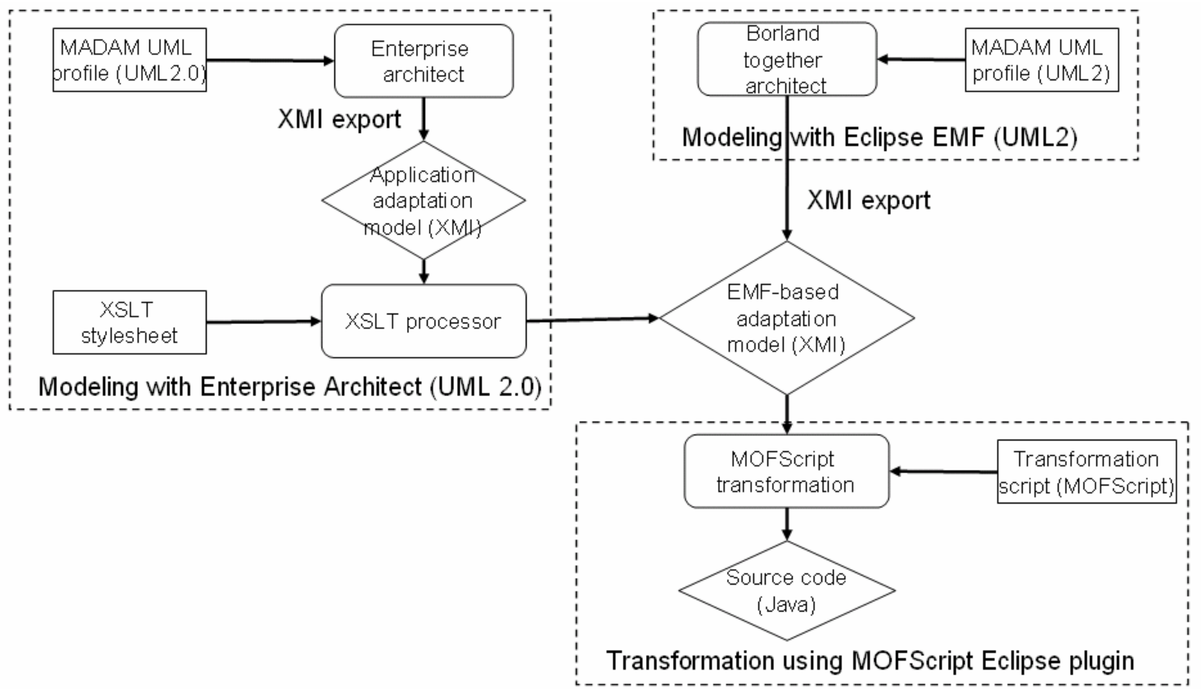
MADAM UML profile (UML2.0)
Enterprise architect
XMI export
Application adaptation model (XMI)
XSLT stylesheet
XSLT processor
Modeling with Enterprise Architect (UML 2.0)
Borland together architect
MADAM UML profile (UML2)
Modeling with Eclipse EMF (UML2)
XMI export
EMF-based adaptation model (XMI)
MOFScript transformation
Transformation script (MOFScript)
Source code (Java)
Transformation using MOFScript Eclipse plugin

Figure 3: The realisation of the MADAM tool chain

6.4 Adaptation Middleware

Figure **4** shows the architecture of the adaptation middleware in terms of its main components and the interactions among them.

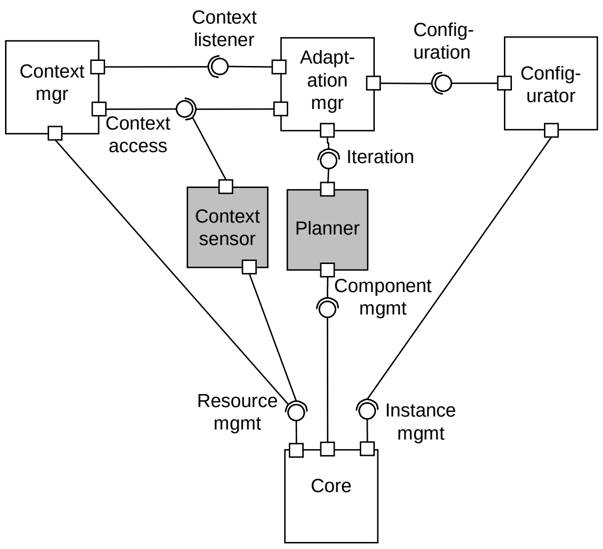

Figure 4: Middleware architecture

The *Core* abstracts the underlying execution platform and component middleware and provides services for Component Management, Instance Management and Resource Management.

- *Component management* supports publishing new component implementations (atomic components or compositions), and retrieving all published implementations of a given component type during application launch or adaptation. Since component retrieval is done at runtime, new component implementations will be considered as soon as they are published and thus dynamic evolution of applications, both in terms of functional and adaptation capabilities, is supported.

- *Instance management* supports controlling components life cycle such as *instantiate, remove, bind, unbind, setting parameters,* and *start and stop* of component instances. This is made possible by reifying the structure of components that is plugged into the component framework. These components can be either application components or context components (i.e. context sensors and context reasoners).
- *Resource management* reifies the underlying computational resources by providing a uniform resource model for access and monitoring. This service also supports resources discovery, which is useful for mobile applications where resources can be discovered and added dynamically just like any other component.

The *Context manager* represents the active part of the middleware architecture. It is responsible for managing and monitoring a set of contexts in the system environment relevant for the adaptation. Context includes execution platform context elements such as network and memory resources, the environment context elements such as light and noise, and user context elements such as location and stress level. This information is collected, represented, and stored using *Context sensors*. There are three main kinds of context sensors: context probes that senses context directly, context reasoners that aim to aggregate, predict and derive new context information, and resource sensors that uses the resource management services of the core to monitor the resources of the execution platform. Context elements are delivered to the Adaptation Manager component when appropriate.

The Adaptation manager is responsible for reasoning on the impact of context changes on the application, determining when there is a need to trigger adaptation of the application, and for selecting an application variant that best fits the current context. In this process, it uses the *Planner* to find the set of possible application variants. The Planner generates this set by composing implementation plans for variation points discovered through the component management interface. When the Adaptation Manager has selected the best adaptation alternative from the set of possible application variants, it then uses the Configurator to implement the new configuration. The adaptation management is described in more detail in [13], where we focus on how architecture models can be exploited to reason about adaptation.

The Configurator is responsible for the configuration and reconfiguration of an application. By reasoning on the difference between the runtime architecture model of the currently running application variant and the model of the new variant, the Configurator is able to derive the necessary configuration steps to get to the new application variant. It uses instance management to create, bind and remove component instances.

The adaptation platform should be deployable on different computers and operating systems and be able to serve applications from different application domains. This requires variability also in the platform. Therefore the platform itself is also built as a component based system family with explicitly modelled variability. This allows applications to extend and customize the platform to suit their needs and it eases the porting of the platform to various computers and operating systems.

Applications may provide their own sensor and reasoner components to cater for application specific context dependencies and they may provide domain specific planners to allow for application specific adaptation logic.

The adaptation middleware implementation, performed in WP4, led by HP and with the participation and contributions of the all the consortium partners, had the goal to fulfil the following objectives:

- Implement the middleware prototype as a vehicle to house the generic runtime support for adaptation proposed by the project.

- Implement the adaptation manager as a generic middleware service exploiting the reconfiguration capabilities of the platform, governed by generic and application specific rules.
- Implement the other adaptation services, such as context monitor and reconfiguration manager as specified in the architecture WP.

These objectives have been achieved through the whole duration of the project with the implementation of three different versions of the middleware, together with the accompanying detailed documentation:

1. A very early proof-of-concept implementation of the application and middleware architecture was completed in month 6, and convinced us about the feasibility of the proposed architecture.
2. The proof-of-concept implementation was extended to provide the initial (1st phase) version of the middleware. A process for distributed software development has also been set up, with appropriate tooling for build, configuration management and bug tracking. This official 1.0 version has achieved good feedback by the pilot services developers, enabling the delivery of 2 different pilot applications.
3. In the second phase of the project WP4 has refined the initial results achieved in the first phase based on the feedback from phase 1 implementation, from architecture work-package and on the experience gathered during pilot service experimentation. This has led to the release of version 2.0 of the middleware software (the final one), that has been improved and enhanced with many new features with respect to version 1.0, that can be summarized in:
    a. *Distribution of the middleware*: distributed resource and context management, planning of distribution of components, remote binding of components, distributed reconfiguration.
    b. *Multiple applications management*: Unified resource allocations across applications, with launching, stopping, and calculation of combined utility, configuration of multiple applications.
    c. *Refined the adaptation strategies*: support for plan variants, planning and allocation of remote reservable resources.
    d. *Architectural constraints support*: in the plan data structure.
    e. *Refined Reconfiguration process*: device and component parameter setting.
    f. *Enhanced resource management*: global-distributed and local resource management in the adaptation domain; lock, reservation and release of resources; added new managed mobile devices resources: microphone and speaker; integration of the network API.
    g. *Enhanced context management*: within the distributed context management: dynamical discovery of remote nodes and context changes propagation in the adaptation domain.
    h. *Plug-in architecture enhancement*: dynamical add and remove of context sensors.
    i. Enhanced the *open-source network API on top of Birdstep Mobile IP* on Windows XP and Windows Mobile
    j. *Control panel (middleware GUI manager) improvements*: adaptation domain distributed computing and context browsing, manual installation of applications, user preferences panel, status bar.
    k. *Optimised performances*.

    Middleware version 2.0, deployed on three different target devices, has also achieved good feedback by the pilot services developers, enabling the delivery of two enhanced different pilot services leveraging the new introduced middleware features.

7 The Pilots

7.1 The Service Technician application

To demonstrate the suitability of the MADAM Middleware, a set of pilot services for mobile users have been developed in the Project. Condat has implemented a pilot derived from a solution for mobile service technicians. The technicians employ mobile devices to organize their tasks, maintain equipment and connect to other servers via GSM, WLAN or Internet. The devices need various adaptations in order to select the optimal network connection or user interface according to the current environment.

The scenario allows validating the full potential of the MADAM middleware in a distributed environment. All main concepts of the adaptation framework, such as context switches, plans, blueprints and adaptations have been tested in many combinations and show the expected, reasonable reactions. The performance measurements for the resulting system perceived by the end users showed good results in all distributed configurations. Only minor improvements, such as for the handling of very many, e.g. more than 1000 adaptation variants, have been proposed.

The development of MADAM Pilot Services can be faced either directly using the MADAM Middleware interface or using the assistance modelling tool set. The tools have been proved to ease the application development on the bases of a small, clear set of model elements. It allows to define the application components, their distribution, the adaptation behaviour and the automatic code generation for the application basis. The results of the evaluation reveal that the objectives of the project for development and modelling facilities are fulfilled.

7.2 The SatMOtion application

7.2.1 The original SatMotion System

SatMotion is a distributed application that implements a tool for setting up Internet via Satellite terminals (also known as VSAT or SIT terminals), providing assistance to the field installer on the antenna alignment procedure.

SatMotion system encloses several applications which offer different operating modes. The major modes are TWO-WAY and ONE-WAY. The TWO-WAY mode implements a two way communication tool which receives satellite signal traces from the SatMotion server and can also send commands to server side in order to change the measurement setup of remote instrumentation, e.g. the spectrum analyzer. SatMotion ONE-WAY is a simplified version of the former mode, enabling just one way communication for the reception of traces from the server. This mode is typically used by the VSAT installer to align the antenna, since no interaction with the measurement server is usually needed in this procedure, and it provides higher information rate than the TWO-WAY mode. In the ONE-WAY mode, the installer starts its line-up by setting the VSAT to transmit a test signal, while obtaining on its handheld a real-time feedback on its actions in the form of signal traces sent remotely from the server.

The above described modes are complemented by an *offline* mode named BackPlayer. This is able to play, perform measurements, generate reports etc. on recorded spectrum activity, received previously in an *online* mode (either ONE-WAY or TWO-WAY) and stored on the handheld.

A hands-free mode is also provided by this application, selected by the user to listen to the measurements in order to use his hands to move the antenna: For the objectives of the project, this new feature will be useful to demonstrate distributed adaptations and migrating the “Text to Speech” component when there is lack of memory; and device configuration by adjusting the speaker volume with respect to the environmental noise. In this type of adaptation, the *company server,* represented in Figure 5, offers support to the client to execute the heavy components of the application “Text to Speech”

Figure 5 depicts the typical SatMotion operational scenario, where measurement instrumentation (1) is connected to one or multiple antennas (2) by means of a switching matrix (8). The antennas are receiving signals from different satellites (3). The instrumentation consists of signal measurement analysers, such as spectrum analysers or vector analysers that receive the signals captured by the antennas. The measurement instrumentation is controlled by a server (4), which at the same time communicates through a wireless network (5) with one or more mobile terminals (6), feeding them with signal traces information..

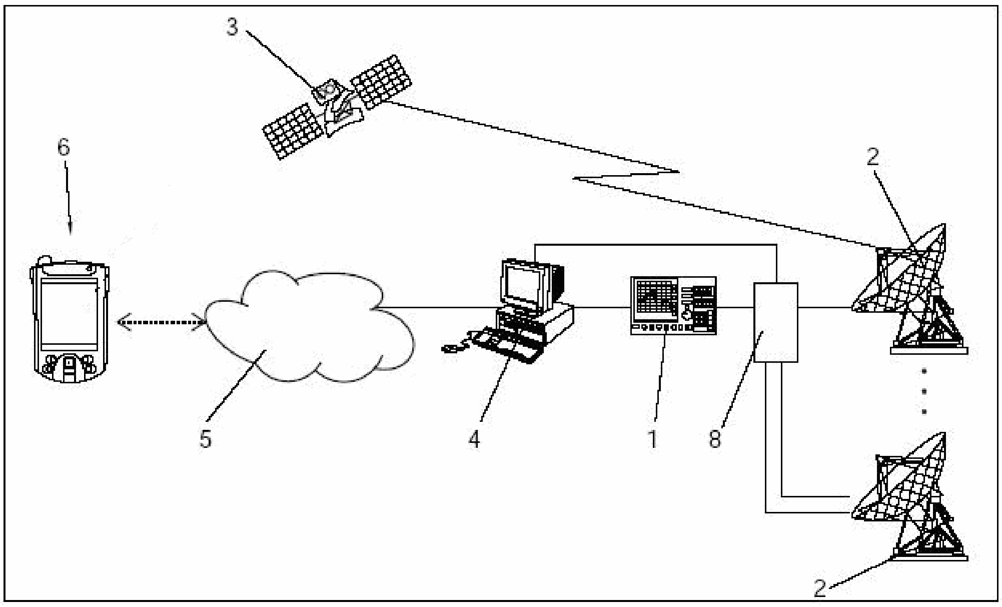

Figure 5: SatMotion infrastructure

7.2.2 The adaptive version of SatMotion

In the original SatMotion version, the described modes were independent applications modules manually selected by the installer in different situations. The MADAM version integrated all the modes into one application, so that the running mode is automatically selected by the MADAM middleware based on installer preference and context conditions.

In addition, we have added a new company server offering support to the client: This server belongs to the installer company and provides distributed processing resources to the clients, allowing the release of resources in the local devices. This new node, together with the SatMotion server, is used to demonstrate the distributed adaptation. Figure 6 depicts the new system infrastructure for the adaptive SatMotion.

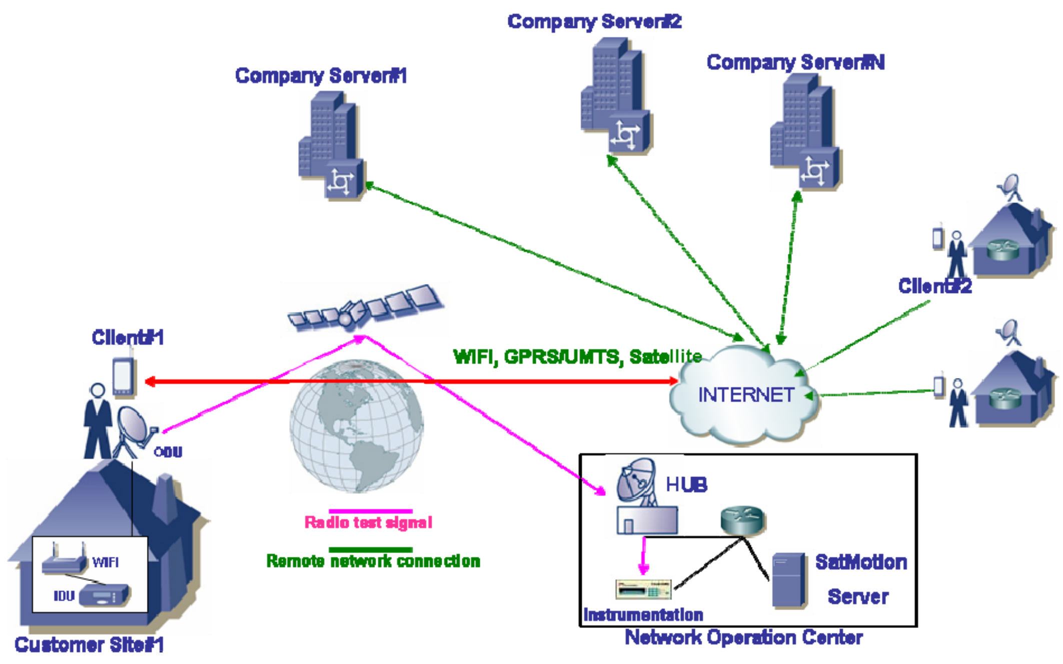

Figure 6: The new system infrastructure for the adaptive SatMotion application

The main context elements whose variations could influence the usability of SatMotion, and therefore trigger the adaptation process, are: power level, available memory, environmental noise, available networks (WLAN, LAN, UMTS, GPRS), network QoS (throughput and latency), network profile (preferred network interface, network power consumption, network economic cost), user preferred mode (operational mode). In addition, in order to carry out distributed adaptation, the middleware should be aware of available remote nodes. This context information is handled by the middleware and is fully transparent to SatMotion application.

The main adaptations implemented in the MADAM version of Satmotion are:

- All SatMotion operation modes have been integrated and are handled by MADAM
  - Are selectable by the end user
  - Are adapted to context conditions such as network, memory and battery conditions and could be automatically forced by MADAM.
  - The state is preserved at mode change
- SatMotion is fully adapted to the networking environment, handling related configuration without user-interaction. This greatly eases installer use as the installer is not usually familiar with networking technology.
- Signal recording operations have been improved in memory-constrained devices: only possible if enough space available.
- A hands-free mode has been added. In a low resources situation, the Text to Speech component is deployed in other MADAM node, unlocking used resources in the local device. This is a distributed adaptation
- Adjusting the speaker volume to the environmental noise.
- Component parameterization depending on context situation, to lock certain functionalities or to control the resolution of the received streaming.

7.2.3 Development process: Adding adaptation to SatMotion

In this section we explain the general process that has been followed by Integrasys in order to develop SatMotion adaptive application on MADAM. This process is derived from the recommendations of

Work Package 3. Please note that this section uses MADAM technical terms, so it is necessary to be familiar with the MADAM approach.

<u>**External tools**</u>

For the development of the SatMotion Pilot Services, we have used the following tools:

- **Enterprise Architect**: Enterprise Architect is diagramming commercial software developed by Sparx System[2]. It has been used to draw the UML diagrams presented in D5.3, although some of them have been slightly modified in order to be adapted to the middleware.
- **Eclipse 3.1**: Eclipse is an open source software development project dedicated to providing a robust, full-featured, and commercial-quality industry platform for the development of highly integrated tools. It has used as a Java Integrated Development Environment:
    - Project management: Create new projects, organisation, setting properties (classpath, external libraries…)
    - Implementing Java code needed for the SatMotion application, compiling and building the application.
    - Debugging tool: Very powerful and useful debug tools have been used in the development process.
    - Exporting the pilot services to a jar file.
    - For the generation of platform specific code, based in MOFScript and Eclipse-EMF. For this purpose we extend Eclipse with the following plug-in: EMF 2.1.0, UML2 and ANTLR.
- **Apache-Ant 1.6.5**: Apache Ant is a Java-based build tool. It has been utilized to build and execute the MADAM platform.

<u>**Initial adaptation analysis**</u>

The first step in the development process consists of the following analysis tasks:

- *Detecting situations where adaptation could be useful*: We analyse how we can improve the user experience adapting the application to certain context situations.
- *Identifying context elements*: Derived from the previous analysis, we extract the context elements that influence the application. We analyse both the needed context information and the context changes that can lead to a loss of quality of service susceptible of being improved through adaptation mechanisms.
- *Refinements of reconfiguration*: We identify the specific pilot service reconfigurations that improve the user experience.

<u>**Modelling the adaptive application**</u>

Component type realization diagrams: The component types have one or more possible realisations. The details of these realisations are defined in later steps. The following figure illustrates the class diagram for the “Controller” component, detailing its different realisations. It can be seen that the Controller component has three atomic realizations, OneWayController, TwoWayController and BackPlayerController.

[2] http://www.sparxsystems.com.au/

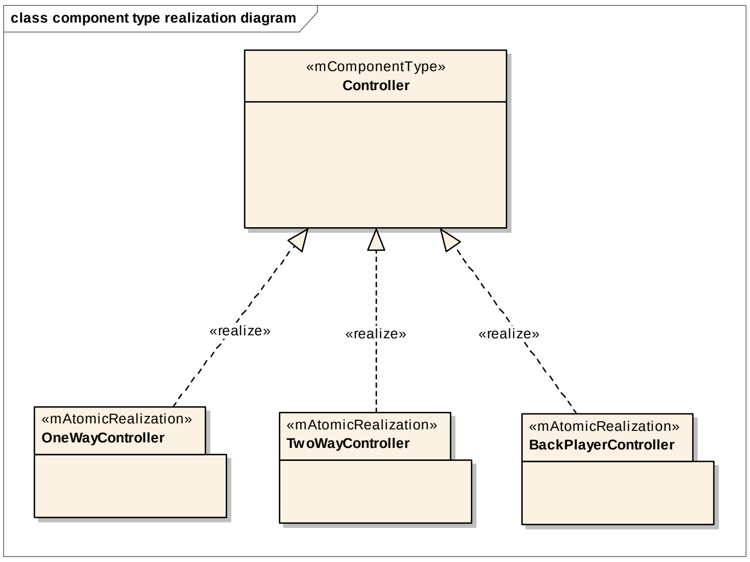

Figure 7: Component Type realization diagram for SatMotion

Feature model diagrams: This is an optional step required if we apply architectural constraints to our application. In this step we define the features and model its hierarchy. The feature hierarchy model of the SatMotion application is depicted in the diagram of Figure 8.

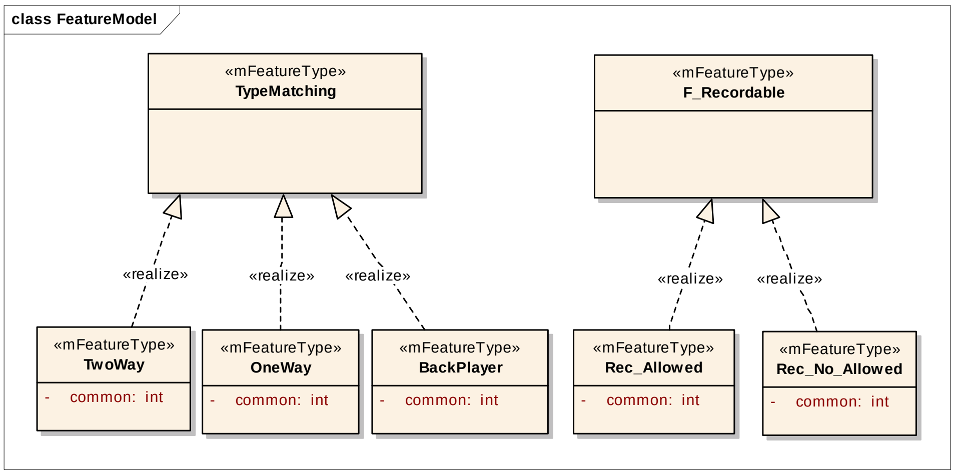

Figure 8: Feature model diagram for SatMotion

Role definition: The following Figure 9 describes the relations between the roles composing the application and the existent component types:

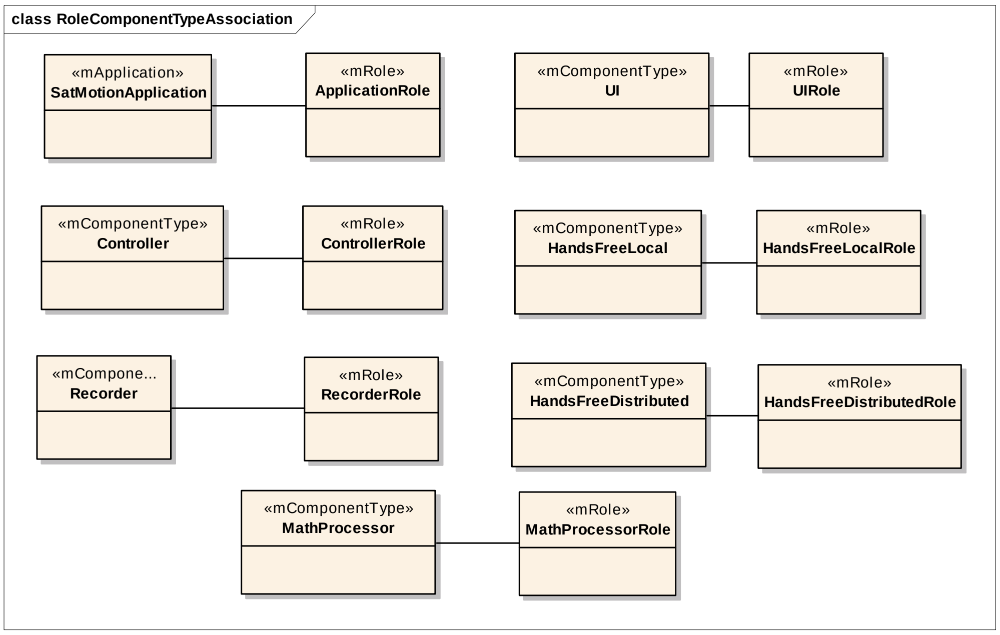
class RoleComponentTypeAssociation
«mApplication» SatMotionApplication
«mRole» ApplicationRole
«mComponentType» UI
«mRole» UIRole
«mComponentType» Controller
«mRole» ControllerRole
«mComponentType» HandsFreeLocal
«mRole» HandsFreeLocalRole
«mCompone... Recorder
«mRole» RecorderRole
«mComponentType» HandsFreeDistributed
«mRole» HandsFreeDistributedRole
«mComponentType» MathProcessor
«mRole» MathProcessorRole

Figure 9: Association roles and component type in SatMotion

Context and resource modelling: We create the diagrams for the resources and context element that influence our application, and whose change can trigger adaptation process in the pilot services. The next diagram of Figure 10 describes the modelling of the properties of the "SoundSystem" resource:

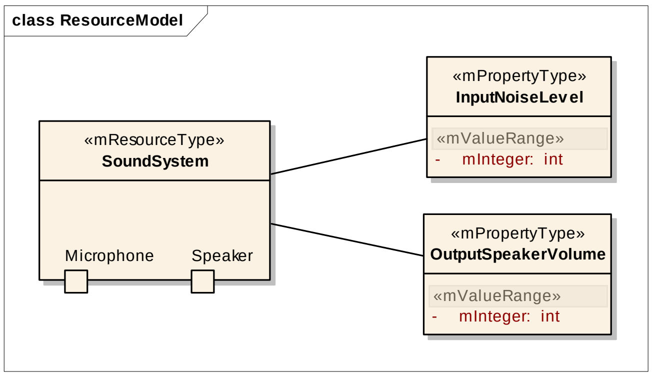
class ResourceModel
«mResourceType» SoundSystem
Microphone
Speaker
«mPropertyType» InputNoiseLevel
«mValueRange»
- mInteger: int
«mPropertyType» OutputSpeakerVolume
«mValueRange»
- mInteger: int

Figure 10: Sound System modelling

Modelling the detailed realisation of the components: The realisation of each of the packages defined in the component type realization diagrams has to be detailed. As an example, Figure 11 is a diagram which depicts the detailed realization of the TwoWayUI component. This class diagram contains the TwoWayUI component together with its associations, parameter settings and properties.

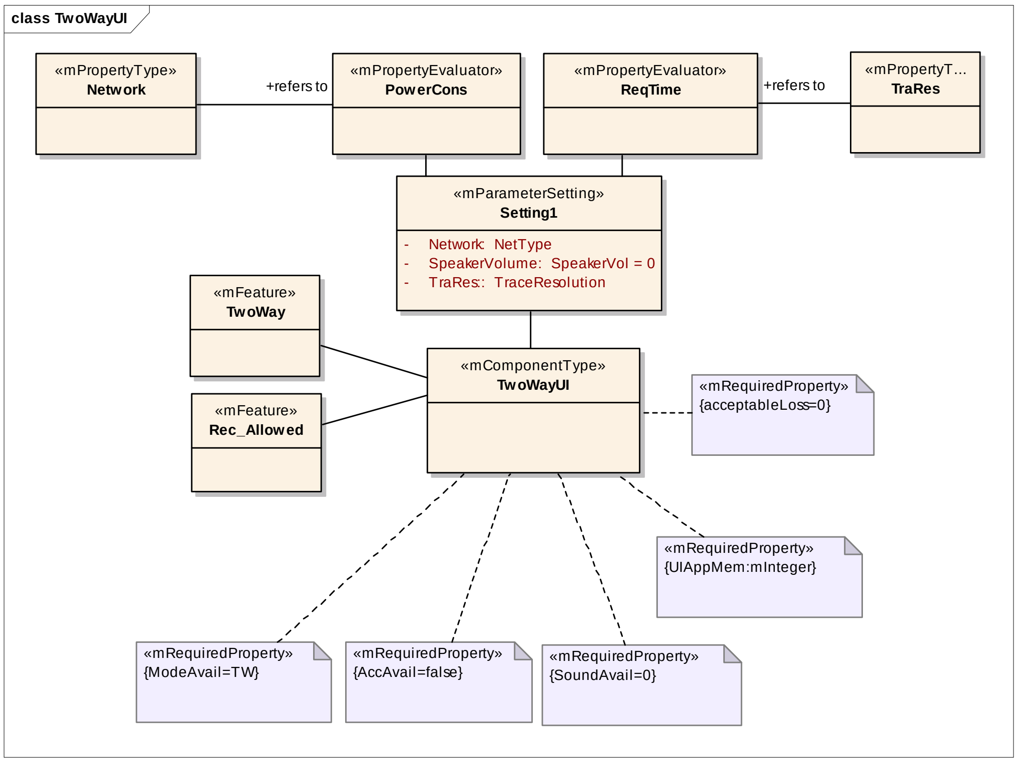
class TwoWayUI
«mPropertyType» Network
+refers to
«mPropertyEvaluator» PowerCons
«mPropertyEvaluator» ReqTime
+refers to
«mPropertyT... TraRes
«mParameterSetting» Setting1
- Network: NetType
- SpeakerVolume: SpeakerVol = 0
- TraRes:: TraceResolution
«mFeature» TwoWay
«mFeature» Rec_Allowed
«mComponentType» TwoWayUI
«mRequiredProperty» {acceptableLoss=0}
«mRequiredProperty» {UIAppMem:mInteger}
«mRequiredProperty» {ModeAvail=TW}
«mRequiredProperty» {AccAvail=false}
«mRequiredProperty» {SoundAvail=0}

Figure 11: Detailed realization of components

Modelling of the realisation compositions: The composite diagram of Figure 12 contains an example of the necessary information for the “SatMotionApplication” component, in terms of the roles, ports and component types involved. Provided that the association between the roles and the components are detailed separately in another diagram, in this one only roles and ports are detailed.

Generate the code

Once we have the model of the adaptive application, we proceed the automatic generation of code. The UML models have been created using Enterprise Architect. We tackle the following tasks:

- We have to transform the UML model into XMI format. For the conversion of the XMI output of Enterprise Architect to the XMI/EMF format an XSLT Stylesheet is provided by WP3. In order to perform this step, the Enterprise Architect offers a user friendly “Export” tool.
- We transform the application adaptation model in XMI to a platform-specific source code with the help of MOFScript and an appropriate transformation script provided by WP3.

Detailed information of how to use modelling tools can be found MADAM D3.4, section 5.4.

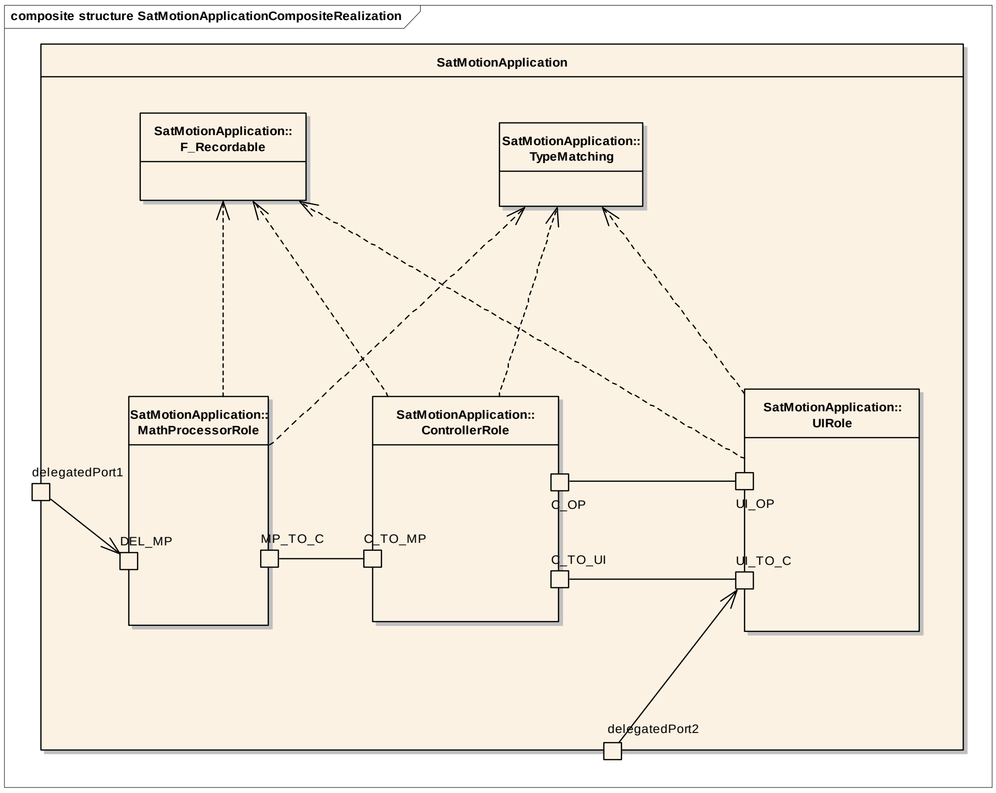
composite structure SatMotionApplicationCompositeRealization
SatMotionApplication
SatMotionApplication:: F_Recordable
SatMotionApplication:: TypeMatching
SatMotionApplication:: MathProcessorRole
SatMotionApplication:: ControllerRole
SatMotionApplication:: UIRole
delegatedPort1
DEL_MP
MP_TO_C
C_TO_MP
C_OP
UI_OP
C_TO_UI
UI_TO_C
delegatedPort2

Figure 12: Realization of a Composition in SatMotion

<u>Complete the generated code</u>

The code provided by modelling tools has to be completed by the application developer, specially the part related to *property evaluators* and *utility functions.* The code generator produces most of application meta-information which is required by the middleware. The meta-information is contained in a class implementing the IApplication interface. Taking as a basis a complete model of the application, the developer should not need to change any part of the generated code, except for filling in the property evaluators and the utility. The declaration is given in Figure 13.

```
public interface PropertyEvaluator
{
    public static final String UTILITY_PROPERTY = "utility";
    public Object evaluate( ContextElement context,
                            PropertyEvaluatorContext evalContext );
}
```

Figure 13: The declaration of a property evaluator

The code in Figure 14 shows an example of implementation of Property evaluators, where the total memory required by a composition plan is calculated summing up the memory of its internal components.

The rules to select the best application for a given context situation are implemented by the developer in the so called *utility function*. *Utility functions* assign a utility value to each application variant as a function of application properties, context and goals. In that way, the impact of reconfiguration on context is taken into account during the evaluation of the utility for each variant. From the implementation viewpoint, the utility function is a regular property evaluator function, except that it has the predefined name “utility” defined in the PropertyEvaluator interface, and that it should always return a double value in the range 0.0 (unusable) to 1.0 (perfect utility).

```
class OnlineRecorderMemoryRequired extends EvaluatorAdapter
{

 public Object evaluate(ContextElement context, PropertyEvaluatorContext
                       evalContext)
 {
   return new Integer(
       ((Integer)evalContext.evaluateForRole(MEM REQ, context, auiRole)).intValue()
       +((Integer)evalContext.evaluateForRole(MEM_REQ, context,
       aControllerRole)).intValue()
       + ((Integer)evalContext.evaluateForRole(MEM_REQ, context,
       recordRole)).intValue()
       + ((Integer)evalContext.evaluateForRole(MEM_REQ, context,
       mathRole)).intValue());
 }
}
```

Figure 14: Example implementation of Property evaluators

When implementing the utility function, the developer should apply appropriate weight to the properties of the application variant, and combine these in an evaluation towards the current context. The utility function is executed as many times as application variants is evaluated by the middleware. So the developer should take care of its performance.

For SatMotion Pilot Service, in the first part of the utility functions, we discard the application variants which clearly are not candidates. Hence we avoid evaluating the rest of the function and consequently the performance will be improved. For instance:

- If the network is unavailable, any online variant will not be selected, and the evaluation of their utility functions will return the zero value.
- If the user has not chosen the TWO-WAY or the ACCURATE mode, the variants associated to these modes will not be selected. The same occurs with the variants associated to the RECORDER, which can not be selected if the user has not requested to record the signal.
- If a variant requires more memory than offered by the system, this variant will be discarded.
- If the variant is not discarded, we calculate four *intermediate utility values* which assess how the variant satisfies different aspects in the current context situation. The four intermediate values are:
    1. *User requested utility*, which relates the *user selected mode*, which is a context element, with the *mode offered by the variant*, which is a property constraint associated to application variants. It takes the highest value when *mode requested* matches *offered mode*. This value takes into account the user preference.
    2. *Recorder utility*, which takes highest value when the *Record requested* matches *Record available.*
    3. *Time activity utility*: It takes higher values when the time activity offered by the device, which is directly obtained from the battery level and power consumption, is higher than the *required activity duration* of the application variant. This value helps to choose variants that make a safe shutdown of the application.
    4. *Response time utility*: It takes higher values when the *response time* offered by the system, which is obtained from the network QoS, is lower than the *required response time* of the variant. This value helps to discern between ONE-WAY and TWO-WAY modes, since TWO-WAY requires better network quality and consequently lower response time.
- Finally the *utility value* is calculated as the average of the intermediate utility values.

<u>Implementing the adaptive application</u>

Everything that has been modelled and commented above, can be implemented in a Java class whose major tasks is to inform the middleware about the Plans, utility functions and context dependencies. We call this class *SMApplication* class

The developer has to add some additional code to the original application. He has to implement the *adaptive methods* which are defined in MADAM D2.3, and given here with the task that they carry out in SatMotion Pilot Service:

- suspend(IConfigurationCoordinator configurator): suspend a thread, a loop, make invisible a Frame…
- init(): initialise variables for running
- startActivity(): start everything (run and restart a thread, make visible a Frame)
- finit(): dispose a window, destroy a thread, close connection…
- getComponentState(): get the state of a component
- setComponentState(): set the state in a component
- removeConnection(): Remove connections between components
- addConnection(): Add connections between components

We have created an intermediate layer which connects the MW to the real SatMotion original application as shown in Figure 15. With this intermediate layer, we minimize the addition of code in the original application and the final application seems to be clearer.

Finally, we have implemented the specific sensors not provided by MADAM in another package of the application following the specification explained in D4.2. The adaptive application has been developed with the Eclipse development environment. The conclusions of this work are presented in chapter 8.

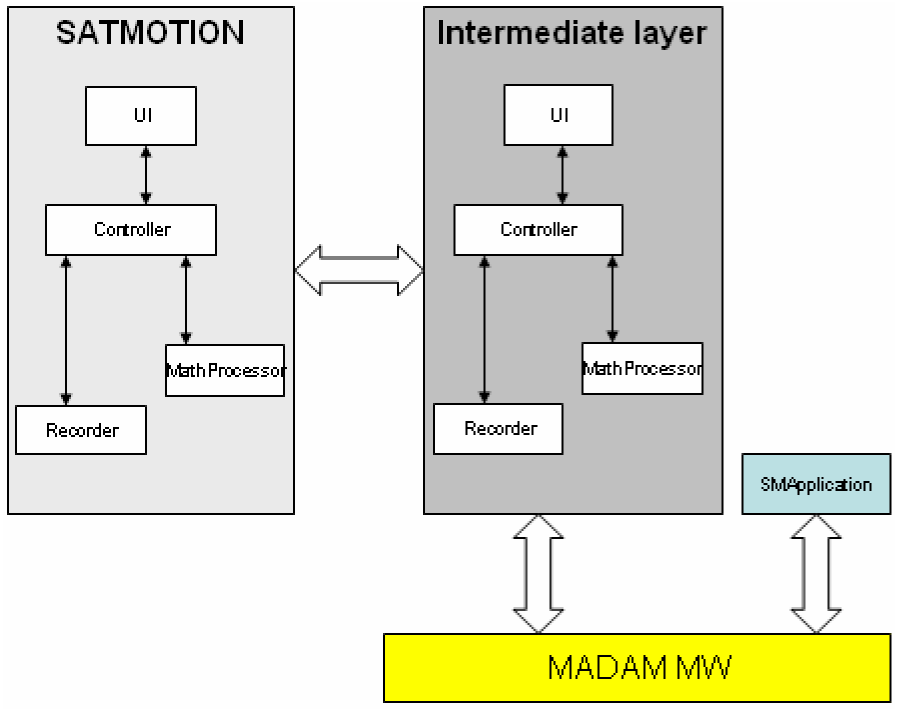

Figure 15: Overall architectural blocks in SatMotion Pilot ServicetMotion Pilot Service

8 Availability of the results

8.1 Scientific impact

During the project period the partners produced 23 papers that have been accepted for publication. D7.1 gives a complete listing. General external dissemination has been achieved by developing a website: www.ist-madam.org. The website is updated at regular intervals and aims at informing the wider audience about the project and its achievements. The source code developed will be distributed as an open source project within one of the available frameworks for this.

8.2 Economic impact

Each partner expects a positive economic impact from the results of the project.

- For SIMULA the project has been an excellent opportunity to apply and extend previous research results at SIMULA, to promote further collaborative research, and for advanced teaching activities. It may also offer an opportunity for SIMULA to contribute to the creation of new businesses.
- Birdstep will integrate concepts developed through the work in MADAM into their software products, and offer these products to operators, enterprises and system integrators.
- Condat expects to exploit the project results by reusing methodology and software components developed within the project to enhance their existing mobile business solutions with dynamic adaptation to changing environment conditions.
- HP will exploit the results of this project in its commercial platform solution portfolio.
- Integrasys will exploit the results on adaptive platforms for further research and to improve the internal software base technology.
- SINTEF's exploitation of the results will be done through the promotion of MADAM platform for specific research applications among the different projects carried out at SINTEF.
- For the University of Kassel the project will present an unique opportunity to apply and extend previous research results in order to solve challenging new research problems. This will foster the development of powerful tools and software platforms that offer opportunities for take-up by industry partners, for further collaborative research, and for advanced teaching activities.
- The Univeristy of Cyprus intends to adapt middleware platforms for mobile devices and to consider quality of service aspects for specifying a good "quality" in the computing environment under different modes of adaptation.

8.3 Impact on standards

No major direct impacts on standards are registered. However the collaboration between European partners is very relevant to define and promote standards that can be used globally. Several aspects of the results of MADAM are relevant for standardisation, for instance

- the interface to the execution environment to collect information about changes,
- extensions to modelling languages to support the specification of adaptation capabilities,
- extensions to component standards to accommodate adaptation oriented information in components.

8.4 Impact on policy

The project does not consider the development to directly influence policy.

9 Lessons learned

Trough the development and testing of the pilot services we have collected experience with the use of the MADAM development framework. In summary we can say that MADAM offers to developers a rich set of features to add adaptation to applications, both related to modelling notations and tools,and to the supporting middleware that leverages the models to implement context monitoring, adaptation reasoning and reconfiguration at runtime.The model-driven development approach has proven to be a viable and effective methodology to aid the software developers to build adaptive applications. Regarding the methods provided by the middleware to developers, we note that they provide intuitive, user-friendly, and flexible interfaces to support adaptation. In the following we discuss in more detail how we have succeeded in satisfying the requirements listed in chapter 2:

- **Provide benefit to user**: It is our clear impression from the experience with the pilot applications that that the self-adaptive behaviour implemented by means of the MADAM framework benefits the users. However, we have not been able to verify this by means of systematic studies invoving real users.
- **Limit developer burdens:** Regarding the development process we discussed the extra effort needed in general to endow an application with adaptation capabilities, and the overhead effort in the MADAM case as opposed to the use of *ad-hoc* techniques:

  Required extra effort to enable adaptive applications: This effort will obviously depend on the scope of the wanted adaptations and the complexity of the applications. It mainly accounts for the implementation of needed variants and any pre-requisite of the adaptation framework, e.g. complying with component frameworks in the case of MADAM. In addition, applications handling critical data or operations may also require complicate procedures for state-transfer between variants during reconfigurations.

  For example, most of the variants of the SatMotion Pilot Service were previously implemented as different applications used in different situations. The re-design of SatMotion in MADAM has resulted in important use enhancements that justify the development effort incurred.

  In general, we think that for applications with a substantial degree of functional-complexity and context-sensitivity, the extra effort is easily justified considering the benefits that can be obtained. For certain applications, e.g. those requiring high usability in a wide range of context situations, including those with strongly adverse conditions, it appears that adaptation is a must rather than a choice.

  - *MADAM effort vs. ad-hoc effort*. We compare the effort of developing an adaptive application with and without MADAM support, regarding the following situations:
    - The functional complexity is low: In this case, the advantages of using MADAM in its current status seem to be rather limited and it may be better to use ad hoc techniques, which can improve the runtime performance as a side effect. On the other hand, MADAM could increase the utility for this kind of simple applications offering intuitive reconfiguration adaptations through basic and easy to use mechanisms. We refer to adaptations such as adjusting speaker volume in noisy environments, adjusting the brightness according to environmental light, selecting the most appropriate network, or activating cache mechanisms in reaction to predicted network disconnection.
    - Applications with low sensitivity to context. Consider the case of mobile applications deployed in stable operational environments, e.g. those where just one network type is available with assured and non-modifiable QoS, plenty of power resources and unchanging environmental conditions. Application changes are mainly of functional nature and driven by end-user input. Nevertheless, if

these changes could also be triggered in response to some context change, the developer may want to add some adaptation, and the best choice in this case appears to be the use of ad-hoc techniques.

- o The functional complexity is high. MADAM offers general and powerful mechanisms which facilitate the development of adaptive applications for this case. Even if context sensitivity of the applications is not particularly high, MADAM provides valuable mechanism to automate application procedures including device configuration, and hiding complexity to end-users. The assignment of property constraints to plans, the access to context information, the flexibility to implement rules for best variant selection, makes the MADAM framework a very good solution to empower complex applications with adaptation and self-reconfiguration capabilities, offering unified and reusable mechanisms to the developer.

- **Limit runtime overhead:** Regarding the evaluation of MADAM in run-time, we have tested both functional and extra-functional aspects of the platform. The functional tests demonstrate the correct functioning of the adaptive application on top of the MADAM middleware.

  Regarding the resource consumption, we concentrated on measuring the *memory* allocated by MADAM. The result was between 6 and 8 MByte for a PC and a PDA with their different operating systems. This meets the specified criteria for the PC and is therefore acceptable. Only for the PDA this result is at the limit of the defined criteria and should therefore be improved.

  The time needed to carry out the adaptation, is a critical issue that impacts the user acceptance, since while the adaptation is being performed, the application will not run. The results are good for the PC, although the time strongly depends on the specific reconfiguration performed. This time is very dependent on the application. For SatMotion in a PC the reconfiguration time fluctuates between 0,5 and 1,2 seconds. On the PDA, the measurement results are in the limits of acceptance. In this case the reconfiguration time fluctuates between 4,5 and 6 seconds.

  Despite the fact that these measurments indicates acceptable runtime overhead also on the PDA, we observed very poor performance when running the pilot applications on the PDA used for the performance measurments. We believe that the reason is that the middleware and the application together filled up the memory on the PDA to such an extent that frequent garbage collection slowed down the execution significantly, both of the application itself and the middleware.. Some informal tests on a newer PDA model exhibited much better performance. Therefore we believe that with more powerful handheld devices appearing on the market and a more optimised implementation of the middleware, it will be possible to support adaptation support on handheld devices running multiple applications with acceptable performance.
- **Component based:** In our approach the architecture model is built from part models associated with each component, allowing that the additional artefacts necessary to support self-adaptation are developed in a modular way. The models of composite components in effect describe sub-frameworks where each role defined can be played by any component matching the component type defined by the role. This allows an extensible set of components to be used as part of the application framework.
- **Exploit reuse**: In the MADAM solution, a major part of the logic needed to achieve self-adaptation comes in the form of generic middleware services. These middleware components

are generally reusable across application domains, and therefor have the potential to reduce the need for application specific programming to support self-adaptation significantly.

- **Separate concerns**: The MADAM modelling notation allows for a clear separation of the aspects related to adaptation from the aspects related to functionality, allowing the developer to focus on one adpect at the time.

The Madam technology is still in a prototype stage of maturity and the pilot application development has proposed a set of ideas that could improve the usability of the MADAM platform. The main suggestions are:

- Standard mechanisms to manage system failure, which are not currently provided by the framework.
- Avoid the selection of meaningless component combinations, not directly supported by the platform. The modelling notations offer the means to specify architectural constraints, but this is not fully implemented by the middleware.
- Automate state preservation between consecutive reconfigurations. Some preliminary mechanisms are proposed, but most of the responsibility is left to the developer.
- Support to extend the configurable resources. The current approach just allows the selection of the network interface and adjusts the speaker volume. An extensibility mechanism that allows the developer to reconfigure other device resources would be appreciated.
- Improved handling of parameterised components. The current approach is intuitive and works fine, but it does not allow the use of continuous values.
- Mechanisms are needed that allow the application to connect to remote non-MADAM components, such as web services. The current implementation just allows binding remote components that have been previously deployed on a remote node. At runtime, we assume that the slave nodes provide the required remote components, but there are no mechanisms to dynamically discover these components.
- Security mechanisms are needed to protect the user against intruders who intend to interfere with adaptations in malicious ways.

It can also be mentioned that more practical experience would be needed to fully evaluate the effectiveness of the model-driven approach. Especially, the suitability for more complex applications needs to be investigated further. In addition, developers would like to have even more documentation and illustrative examples about the model-driven methodology than provided in the additional “MADAM How To Guide”. For example, it would be helpful to have further information on

- the derivation of the MADAM model from an existing application and
- the generation of code for the different model elements.

Most partners of the MADAM consortium are engaged in a follow-on project, called MUSIC, where the insights and open challenges of MADAM will be brought forward in order to come up with more general solutions for self-adaptive applications in complex ubiquitous computing environments.

10 The Consortium

10.1 The consortium

The roles of the partners are stated in the following table.

Participant			**Role**	**Contribution Area in MADAM**
SIMULA	1	N	Research	Component frameworks Middleware core platform development Scenarios definition and requirements identification
BIRDSTEP	2	N	Technology	Middleware development with focus on mobile communication Scenarios definition and requirements identification
CONDAT	3	D	Technology and Service provider	Middleware development with focus on mobile business process support Pilot Service development, trial and evaluation Scenarios definition and requirements identification
HP	4	I	Technology	Middleware development Scenarios definition and requirements identification
INTEGRASYS	5	E	Technology and Service provider	Pilot Service development, trial and evaluation Scenarios definition and requirements identification
SINTEF	6	N	Research	Project management Adaptation theory Modelling and methodology support Scenarios definition and requirements identification
UniK	7	D	Research	Modelling and methodology support QoS engineering Scenarios definition and requirements identification
UCY	8	CY	Research	Dynamic reconfiguration Middleware development Scenarios definition and requirements identification

SIMULA Research Laboratory - Department of Networks and Distributed Systems
SIMULA Research Laboratory in Oslo (Norway) is a non-profit research centre that performs basic research, provides educational programs at master and PhD level, and creates new businesses within selected areas of information and communication technology. SIMULA Research Laboratory (SRL) has currently a staff of about 70 researchers. The centre is financed by the ministry for education and research. The *Department of Networks and Distributed* Systems has a staff of 22 researchers. The research area of the Department is Quality of Service (QoS) management for future distributed applications and services. The long term goal of the department is to determine the next generation of QoS enabled architectures and platforms for open, dynamically adaptable distributed systems.

BirdStep Technology ASA
Birdstep Technology ASA, Oslo, Norway is a SME, which today counts about 60 employees. Birdstep is an R&D organization, where most employees are engineers and scientists with MSc or PhD degrees in Computer Science. The company develops technology within three families of products; database managers, wireless access software and Mobile IP solutions. The technology developed in Birdstep is based on more than 10 years of university research and industrial experience by key technical personnel in the organization, and the company is cooperating with leading European, Asian and US telecom operators, as well as equipment manufactures such as Cisco, Nortel, Lucent, Motorola and Ericsson.

CONDAT AG
Condat AG in Berlin (Germany) couples state-of-the-art technology and measurable business benefit. Condat integrate internet, IT systems and wireless communications technologies to “mobilise” data, business processes and organizations. Condat consulting services are geared towards the business management and the organisational implementation of new solutions. We develop integrated solutions based on industry standard software enhanced with Condat platform technology. We offer end-to-end test systems for GSM, GPRS and UMTS to operators and manufacturers of wireless networks. These test systems automatically measure, analyse and simulate network load and quality and support maintenance and control activities.

Hewlett Packard Italiana SRL
HP is a leading global provider of products, technologies, solutions and services to consumers and businesses. The company’s offerings span IT infrastructure, personal computing and access devices, global services and imaging and printing. Our $4 billion (US) annual research and development (R&D) investment fuels the invention of products, solutions and new technologies, so that we can better serve customers and enter new markets. We invent, engineer and deliver technology solutions that drive business value, create social value and improve the lives of our customers.

Integrasys S.A.
Integrasys S.A. is a Spanish SME that provides software and engineering solutions for monitoring telecommunication signals, offering a wide range of products in measurement automation, satellite signal and RF monitoring. Our expertise includes distributed software development technologies (SNMP, CORBA, XML, LDAP), database technologies (SQL, Informix, JDBC, ODBC) and software engineering (ASN.1, SDL, UML). Integrasys has a solid experience in software development in embedded environments, including the implementation of commercial products for mobile phones and PDAs, which apply the latest distributed technologies. During the 4th and 5th frameworks, Integrasys S.A. has participated in seven research European projects as software monitoring, telecom interoperability and test beds developer and systems integrator.

SINTEF ICT

SINTEF ICT (Norway) performs contract research and development for industry and the public sector, based on key technology areas within software engineering, telecommunication, electronics and acoustics. R&D activities range from basic research through applied research and development to commercialisation of results into new products and business ideas, for both the domestic and international markets. SINTEF ICT has vast experience in EU-funded research projects.

The University of Kassel (UniK)

The University of Kassel in Kassel, Germany, is a medium-sized university that offers a wide spectrum of subjects including engineering sciences (***http://www.uni-kassel.de***). The university has about 18000 students in 14 departments. The chair for Distributed Systems in the Department of Electrical Engineering and Computer Science is headed by Professor Kurt Geihs. Before moving to Kassel in November 2004 he held a chair at the TU Berlin in Germany.

Distributed computing and platforms are the main research themes of the group. Recent research projects have successfully tackled issues such as quality of service extensions in middleware, model-driven design and implementation of distributed applications, the use of semantic web technologies for service-oriented architectures, mobile agents for network and system management, as well as middleware for mobile systems.

University of Cyprus - Software Engineering and Internet Technologies (SEIT) Lab, Computer Science Department

The SEIT lab is part of the Computer Science Department of the University of Cyprus and is directed by Professor George A. Papadopoulos. It undertakes research in the areas of component based software engineering, open and distance learning, mobile computing, parallel and distributed computing, e-Commerce and bioinformatics. The lab is actively involved in the 5th Framework, where it participates in seven EU funded projects, five IST ones (in the areas of telemedicine, bioinformatics, grid technology, component-based systems) and two more belonging to the EUMEDIS (in Open and Distance Learning) initiative. The Laboratory has also been involved in other EU funded initiatives, notably ESIS (European Survey in Information Society, but also the Leonardo Da Vinci framework on Vocational Training where it has participated in 8 such funded projects.

10.2 Contacts

Simula Research Laboratory Martin Linges v 17, Fornebu P.O.Box 134, 1325 Lysaker, Norway Phone : +47 67 82 82 00 Contact person: Frank Eliassen Email: frank@simula.no	**Birdstep Technology ASA** Bryggegata 7 N-0250 Oslo Norway Phone: +47 24 13 47 00 Contact person: Haakon Bryhni Email: haakon.bryhni@birdstep.com
Condat Informationssysteme AG Alt-Moabit 91 d D-10559 Berlin, Germany Phone: +49 30 39 49 0 Contact person: John Klimasek Email: john.klimasek@condat.de	**HP European Innovation Centre** via A. Grandi, 4 20063 Cernusco sul Naviglio, Milan, Italy Phone: +39 348 8513999 Contact person: Alessandro Mamelli Email: alessandro.mamelli@hp.com
Integrasys S.A. Esquilo, 1 -28230 Madrid, Spain Phone: +34 91 631 68 46 Contact person: José Manuel Sanchez Email: jose.sanchez@integrasys.es	**SINTEF ICT** Strindveien 4 NO-7465 Trondheim, Norway Phone: +47 73 59 30 00 Contact person: Geir Horn Email: geir.horn@sintef.no
University of Kassel FB 16, Wilhelmshöher Allee 73 D-34121 Kassel, Germany Phone: +49 561 804-6275 Contact person: Kurt Geihs Email: geihs@uni-kassel.de	**University of Cyprus** 75 Kallipoleos Street P.O. Box 20537, CY-1678 Nicosia, Cyprus Phone: +357 22 892693 Contact person: George Papadopoulos Email: george@cs.ucy.ac.cy