



Information Technology for European Advancement

REFINING THE COMMON REFERENCE MODEL FOR ADAPTIVITY

Deliverable D10
Public version

User-Centered Intelligence:
BEYOND (the GUI)

Edited by : M. Mulder (Delft University of Technology)

*This document is part of the work of the
EUREKA 2023 - ITEA 99002 project BEYOND.
Copyright © 1999-2001 BEYOND consortium
Published via the Beyond Website:
www.extra.research.philips.com/euprojects/beyond*

No part of this publication may be reproduced, stored or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, without written permission of the BEYOND consortium.

Requests for publications can be send to: al_publicrequest@natlab.research.philips.com.

Contents

1.	GENERAL INTRODUCTION	5
1.1.	The common reference model for adaptivity	5
1.2.	The contents of this report	6
2.	DEVELOPMENT OF THE SECOND PROTOTYPE IN THE AVIONICS DOMAIN	9
2.1.	Introduction	9
2.2.	Lessons learned with the first prototype	9
2.3.	Application of the common reference model	11
2.4.	Open issues and recommendations	12
3.	DEVELOPMENT OF THE SECOND PROTOTYPE IN THE HOME DOMAIN (1)	15
3.1.	Introduction	15
4.	DEVELOPMENT OF THE SECOND PROTOTYPE IN THE HOME DOMAIN (2)	17
4.1.	Introduction	17
4.2.	Lessons learned with the first prototype	18
4.3.	Application of the common reference model	19
4.4.	Open issues	20
4.5.	Conclusions and recommendations	21
5.	DEVELOPMENT OF THE SECOND PROTOTYPE IN THE VETRONICS DOMAIN	23
5.1.	Introduction	23
5.2.	Lessons learned with the first prototype	23
5.3.	Changed software architecture to improve flexibility	23
5.4.	Open issues	25
5.5.	Conclusions and recommendations	26
6.	GENERAL CONCLUSIONS AND RECOMMENDATIONS	27

7. REFERENCES 29

1. General introduction

1.1. The common reference model for adaptivity

User interface adaptivity is one of the most promising ways of improving the interaction between the user and the system. An intelligent user interface gathers information about the current situation, uses this internal knowledge to infer what information and means for interaction are most needed, and can adapt itself to provide this information and interaction means at any given time.

The objective of Workpackage 2 is to study the software technology needed in order to achieve an adaptive user interface. A generic framework for adaptivity, the so-called “Common Reference Model for User Interface Adaptivity” has been developed and reported in detail in deliverable D3. This common reference model is a high-level description of the software components of an adaptive user interface, describing their relations in the adaptive system as well as their functions in the interaction with the user and the environment. It is illustrated in Figure 1. The common reference model has been used in the development of the first prototypes in the domains of home, vetronics and avionics.

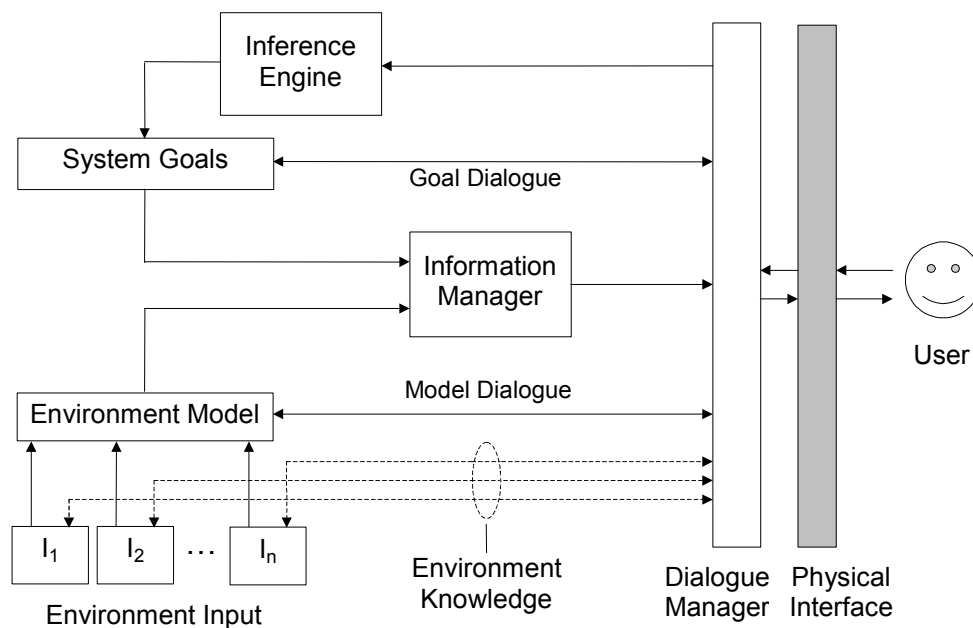


Figure 1 *The common reference model for adaptivity (deliverable D3).*

The common reference model consists of four main components, and each main component consists of one or more sub-components, as will be described shortly below.

The *environment perception component* uses the information about the 'environment' as sensed by the environment input units I_1, I_2, \dots, I_n . It represents the information giving the user interface awareness of context.

The *goal-inference component* consists of the Inference Engine and the System Goals modules. It represents the part of the system that knows or infers the current system and user goals, respectively.

The *information-matching component* consists of the Information Manager. It integrates the knowledge about the environment (the 'constraints') and the user goals (the 'goals'), prioritises the information, and transfers this information to the Dialogue Manager.

The *user-system interaction component* consists of the Dialogue Manager and the Physical Interface, covering all aspects of direct interaction between the user and the system. As this interaction can and in most cases should be multi-modal, here is where Workpackages 1 and 2 have common ground.

1.2. The contents of this report

The purpose of deliverable D10, entitled "Refining the Common Reference Model for Adaptivity", is to collect the experiences gained with the application of the common reference model in the first and second prototypes, and, based on these experiences, to analyse whether the common reference model would need to be altered or re-defined. In this respect, the current deliverable is a second iteration on the definition a common reference model in D3.

During the course of the project, however, partners agreed upon the fact that concentrating purely on the refining of the common reference model would be a too narrow view upon the 'lessons learned' in experimenting with adaptive user interface software technology. The scope of this document is somewhat broader, as is reflected in the outline of this report, stated below.

The withdrawal of the German partners in this project caused Workpackage 4 to be removed, resulting in a lack in across-domains usability tests conducted by experts in the field. In an attempt to compensate for this, the partners have conducted the usability tests themselves, but on a smaller scale than planned and yielding less feedback than expected. This reduced the extent to which the common reference model was evaluated.

In each of the domains covered in this Workpackage, the partners will describe their experiences in the development of their prototypes along the following line of thought. First a short introduction will be given concerning the overall goal of adaptivity in their domain, followed by a description of the main lessons learned in developing adaptive interface technology for the prototypes. Then the common reference model as used in the development of the first and second prototypes will be commented on, emphasising in particular the application-specific differences in the domain covered. The remaining open issues in developing software technologies for adaptive user interfaces will be identified as well as technologies that are emerging that hold a promise for the future.

The document ends with a list of general, i.e. across-domains, conclusions about the results achieved in this Workpackage.

The following people contributed to this document in the domains specified.

Domain	Project	Authors
Avionics	Intelligent, adaptive flight deck	Max Mulder (Delft University of Technology)
Home 1	Content filtering e.g. TV content	Peter Meuleman (Philips Research)
Home 2	Personalised media streams, time shifted A/V	Linde Loomans (Philips Hasselt)
Vetronics	Vetronics UI editor	Johan Devos (BarcoView) Karin Coninx (LUC-EDM)

2. Development of the second prototype in the Avionics domain

2.1. Introduction

Adaptivity can play an important role in making flight deck human-machine interfaces more user-friendly. BarcoView (avionics division) and Delft University of Technology are co-operating in Beyond to develop an intelligent, adaptive flight deck capable of dealing with some of the requirements for future air navigation environments. This chapter describes the development of the second prototype in the avionics domain, with a special emphasis on the implementation of the lessons learned in the usability tests with prototype one regarding the common reference model for adaptivity.

2.2. Lessons learned with the first prototype

2.2.1. Results of usability tests using domain experts

As described in detail in the report written for deliverable D8, the first prototype of the intelligent adaptive flight deck consisted of the implementation of a group of aircraft warning systems, in particular the Ground Proximity Warning System (GPWS, monitoring the separation of the aircraft with terrain) and the Traffic alert and Collision Avoidance System (TCAS, monitoring the separation of the aircraft with other traffic). A multi-agent system architecture is applied that allows these systems to communicate with each other on various levels, resulting in what is called the *warning system layer*.

Integrating warning system information on a system level means that the system becomes aware of the *complete* situation, allowing it to much better inform the crew about what is important at a given time. This in contrast to the current situation where it is the crew who has to integrate the various messages from the warning systems in order to obtain a complete mental representation of the flight situation.

The pilot interface consisted of an advanced tunnel-in-the-sky display showing the pilot a three-dimensional synthetic view of the environment (the Primary Flight Display (PFD)) accompanied by a two-dimensional bird's eye view of the environment (the Navigation Display (ND)). To detect and correct for errors, and to help the pilot in decision making, the warning-system adaptation was implemented in particular on the Navigation Display, including the GPWS and TCAS systems introduced above.

A realistic scenario was developed to test the functionality of the first prototype. Through a fixed (i.e. no-interaction) demonstration a group of over 25 avionics systems experts were confronted with the characteristics of the first prototype, after which they were asked to complete a questionnaire. The main lessons learned were (Steentjes & Mulder, 2000):

- pilots want to remain in charge of their interface as much as possible;
- pilots prefer display elements to be removed completely from their display when that particular element is of no importance to their tasks anymore;
- pilots expressed a need of being able to 'see-through' the automation.

These and other expert comments were used in the development of the second prototype.

2.2.2. Architectural issues

The second prototype development aimed in particular on the design of an *intelligent display layer* on top of the warning system layer. The intelligent display layer is able to adapt the glass cockpit displays, in particular the Navigation Display (ND), to changes in the environment. The intelligent display layer has knowledge on what information *is* currently being presented on the ND (selected by the pilot), and it communicates with the warning system layer on what information *should* in principle be presented. Based on a possible discrepancy between the two knowledge sets, the layer is capable of automatically changing the ND display in such a way that the discrepancy is removed. A logic tree has been developed that determines the best way for adaptation to the various situations that could occur. The systems' adaptivity is being implemented in two ways: by adding or deleting information (terrain, traffic, beacons, airports, routes, etc.) from the display and, second, by making certain elements of the display more 'visible' to the pilot (a change of colour, other attributes or even an automatic change of display range to bring a possible threat into view of the pilot). The intelligent display layer is also based on the multi-agent system approach, to allow for an easy integration with the warning system layer and to implement the required intelligence at the same time.

2.2.3. Multi-modality

The second prototype, as well as the first, is not a multi-modal system. The aural warning messages that usually are generated by the array of warning systems are removed. Instead, the warning systems are integrated to achieve a better, more complete, system internal representation of the environment threats (terrain, traffic, weather, etc.). The crew is informed using an intuitive graphical presentation of the *total* manoeuvring space as determined by the multi-agent, integrated warning system layer. The loss of the aural warning capability is a drawback of the current prototypes. The multi-agent framework is very suitable, however, to incorporate an 'aural warning agent' that can take care of presenting the integrated aural warning to the crew. It were mere timing constraints that prevented the development of this agent, a development that is strongly recommended.

2.2.4. Market relevance

Research into improving the transfer of information between the flight crew and the avionics systems is a crucial step in the development and introduction of new air navigation systems like free flight. The human actors (pilots, controllers) will continue to play an important role in any future system, despite (or one should say, in particular with) the increase of automation in the management of air traffic. All the major players in the avionics domain have integrated a human-centered design philosophy in their product design.

The intelligent, adaptive flight deck interface prototypes as developed in this project are probably among the most advanced avionics human-machine interfaces designed for civil aviation. Although there are also some drawbacks in using the multi-agent system architecture (such as its rather 'slow' real-time-ness) the agent paradigm allowed us to quickly add agents for specific tasks, making the approach flexible and extendable.

2.3. Application of the common reference model

2.3.1. Translation of the reference model to the avionics domain

In the avionics domain the translation of the common reference model to the domain is straightforward, and therefore one can directly refer to Figure 1.

The *warning system layer* is the main part of the Environment Model component. Here the various sensors (for traffic, terrain and weather) measure the data necessary to act as inputs for the warning system logic. The integration of the warning systems in the first prototype yields system awareness, i.e. the system has comprehensive, complete and integrated knowledge of the environment of the aircraft.

The second prototype aimed in particular on the development of the *intelligent display layer*. Since the goals of an avionics system are more or less fixed (safety, fuel consumption, passenger comfort, time constraints, ...) the System Goals component is implemented as a list of goals with for each goal its goal attributes. The goal dialogue and the goal inference have not been implemented because it was considered to be more important to enable the user interface to adapt to *context*. Consequently, all efforts have been put into extending the Environment Model component, and the development of the Dialogue Manager and the Information Manager.

The Environment Model extension comprises the knowledge that the user interface has on the current characteristics of the displays and the type and content of the information being displayed (for instance, it 'knows' that the ND map display has a selected range of 12 nautical mile). The Information Manager uses this knowledge about the current information being presented and the current information needs (as specified by the warning system layer) to minimise the discrepancy between the two sets of information. For example, when there is a traffic warning and the display is currently not presenting any traffic information (as selected by the pilot), the Information Manager will in critical situations adapt the interface in such a way that the traffic is presented. Or, when the traffic is presented but the warning relates to an aircraft that is positioned beyond the displayed range, the Information Manager will automatically adjust the range of the map to include the traffic threat. The Dialogue Manager allows the user to bring up the information underlying the warning system messages, to check and see-through the automation as required by the expert/pilot comments after the first prototype. It also serves as a way to communicate the suggested and automatically conducted changes in the user interface to the crew to give them the feeling that they are still in charge.

2.3.2. Conclusions after the two prototypes

The common reference model suited very well in the development of an intelligent, adaptive flight deck, guiding future research as will be discussed below. Although it is quite abstract in nature and some components definitely call for further research, such as the goal inference, both the first prototype as well as the second prototype fitted well in the model. In fact the evaluations of the first prototype did not call for any changes in the reference model.

2.4. Open issues and recommendations

2.4.1. Reference model parts to be filled in

In the avionics domain, all components of the common reference model have been implemented, except one. The *goal inference component* needs to be fully integrated in the existing architecture, and this remains one of the main avenues for further research. The reason why this component has not been implemented in the two prototypes is twofold. First of all, there were the rather tough time constraints of this project, two years is not much when considering the major breakthroughs that have to be achieved before being able to truly infer what the goal(s) of the pilots are. The second and most important reason is that the goal inference component is simply of less importance in the avionics domain.

As mentioned before, the high-level system goals are crisp and clear, they do not depend on the user at all. Remember that all pilots are well-trained professionals and the need for adaptivity to user-preferences is small. The goals of system safety and efficiency (fuel cost) are always the same, and the most important thing is how the environmental constraints have an effect on the viability of achieving the fixed system goals. It is clear that this aspect is well covered in Beyond. One possible application of a goal inference engine in the cockpit would be to infer whether the crew are doing something themselves that could have an effect on the high-level system goals. In such a situation, the goal inference component could warn the intelligent flight deck that it is not something in the outside world (in the environment) but inside the cockpit that affects the safety and/or efficiency of flight.

2.4.2. Challenges expected for the future

The main challenge in introducing artificial intelligence in the cockpit to provide intelligent, adaptive user interfaces would be the *certification* of the software. Whereas certification of software is already quite difficult in aerospace, the intelligent software is much less predictable which makes certifying it even harder. A trend that is important in this respect is the greater freedom avionics manufacturers have received the past couple of years to improve and update their products.

A second main challenge would be the *acceptance* of pilots of the various levels of authority of the automated systems. As the user-evaluations of the first and second prototypes showed, pilots are not very enthusiastic about the system or user interface to adapt itself automatically. One should keep in mind, however, that pilots are generally quite conservative in their opinions on any changes in aspects of their professional life. It is a well-known fact that when confronted with innovative technology pilots generally tend to be positive on the technology that stands closest to the systems that they have used before.

In our opinion the adaptive user-interface technology will in the long term replace the current flight deck interfaces. Perhaps not to please pilots in the first place with new fancy technology, but rather to safeguard the system goals of efficiency and especially safety. Although there are quite some arguments against it, the common belief that approximately 70 percent of all aircraft accidents can be attributed to human error can not be ignored. On the long term the current philosophy of 'pilot-in-command' could shift gradually to a philosophy that indeed the crew is in command *unless* the crew is doing something that unmistakably could lead to an unaccepted decrease of safety. Ignoring vital information because of 'funneling' their attention towards solving a problem on-board, a common scenario for disaster, should be rendered impossible by introducing technology that intelligently checks

and forecasts the current flight situation, and, if necessary, *acts* on this information.

A second challenge for the current interface development is to incorporate more future-oriented capabilities. In a future ATM environment such as Free Flight, aircraft are much more autonomous and have many possibilities to digitally communicate with each other. The air traffic can then be seen as a multi-agent system in itself, with aircraft as agents and the digital datalinks as a communication service for these agents. The adaptive flight deck should be extended to incorporate the negotiation-based 4D navigation environment of Free Flight. Because it is already based on a multi-agent structure, it is expected to be well-suited for this purpose.

2.4.3. Benchmark

Adaptive interface technology has not emerged anywhere in the cockpits of modern civil aircraft. So in this respect, the current development is unique and benchmarking is impossible. Some things remain to be said, however.

Although with the advent of Multi-Function Display (MFD) technology pilots nowadays have some freedom on what kind of display and/or information they can put on the MFDs, this 'selection' is passive and fixed. Fixed because there is a very limited number of pre-defined selections possible. Passive because the system does not 'do' anything with the information it can gather when intelligently considering the selections the pilot makes. In this respect, the intelligent, adaptive flight deck is truly a novelty that deserves to be developed further.

As explained above, the low level of (one could say, not existing) user interface adaptivity can be attributed to the conservative attitude that most avionics companies have regarding the definition of their user interfaces and also the conservatism of pilots regarding the system they use in their professional environment. From a safety perspective, however, the opinion that automation should in some cases interfere with pilot goal-directed behaviour is gradually gaining ground. In very extreme situations, like a pilot with suicidal tendencies who switches off all automation and controls his aircraft and all passengers into the ground, it is clear that there are limits to the full-authority position that pilots have nowadays. Not pilot authority, but *passenger safety* should be the main focal point in future flight deck design.

The intelligent, adaptive flight deck with its multi-agent system architecture can serve well as the prototyping platform of future flight deck developments.

3. Development of the second prototype in the Home domain (1)

3.1. Introduction

Deleted for the public version

4. Development of the second prototype in the Home domain (2)

4.1. Introduction

The aim of the Philips Hasselt contribution to Beyond “System for personalised media streams: time shifted audio/video”, is to show the feasibility of a time shift function for advanced set top box platforms.

A “basic” set-top box is a receiver/decoder for pay TV services. Platforms for “advanced set top boxes” bring also internet content on the television screen. Besides internet functionality, the advanced set top boxes will support digital audio/video and offer advanced services. Advanced services let consumers take control of their television viewing, by allowing them to watch what they want, when they want it.

Currently available time shift systems support only analogue audio/video. For the new platforms, a time shift function supporting digital television is required.

A constraint in the home domain is that the costs for the solution should be low. Currently available time shift systems use an expensive hardware encoder. In consumer systems a cheaper solution is preferred and we will investigate also software solutions.

To allow time shifting, the new platforms need to be extended with a hard disk. The project also investigates the possibilities for extension of the new platform with a hard disk.

To show the feasibility of time shifted audio/video for advanced set top box platforms we followed the next process steps:

1. requirements for the first prototype
2. analysis of the requirements and architecture description of the first prototype
3. realisation of the first prototype
4. evaluation of the first prototype
5. requirements for the second prototype, derived from the experiences
6. architecture for the second prototype
7. realisation of the second prototype
8. evaluation of the second prototype: feasibility of industrialisation

Section 4.2 gives an overview of the lessons we learned during the first 5 process steps.

As already mentioned before, Philips Hasselt concentrated on the supporting technology for adaptivity in consumer systems and not on the adaptivity of the user interface itself. This is the reason why, at first sight, the relation between the time shift technology and the reference model is rather limited. However, the reference model has given us important insights in the context of adaptivity. This is explained in section 4.3.

Section 4.4 describes the issues, which we could not investigate in the context of Beyond because of time constraints.

Section 4.5 lists our conclusions and recommendations.

4.2. Lessons learned with the first prototype

step 1: requirements for the first prototype

When we started this project, we were very ambitious and technology-minded. We knew the basic features of commercially available time-shift-boxes (like Tivo and Replay), and their technological principles. With this background knowledge, we created an extensive requirements list of the first prototype (deliverable D2).

step 2: analysis of the requirements and architecture description of the first prototype

During the architecture development activities, it became clear that the requirements list was too optimistic, and that some of the requirements listed in deliverable D2 could not be realised. The two main reasons were: hardware limitations of the available IC's, and the absence of the actual application to let the user select streams to store, record or delete.

We also realised that the issue of dynamic partitions is a huge and very advanced file-system issue, not needed to demonstrate basic time-shift, and finally the issue of supporting multiple Hard Disks has been dropped.

step 3: realisation of the first prototype

The main objective of the first prototype, namely to demonstrate "analog time-shift", has been realised. Most of the effort has gone into architectural issues, and the real-time files system and disk scheduler as essential components. This resulted in advanced knowledge on technological aspects of time shift in new set-top box platforms, especially in the domain of a robust real-time file-system and sophisticated disk-scheduling algorithms.

step 4: evaluation of the first prototype

The realisation activities of the first prototype had the following important results:

- the awareness that we could not realise all requirements in the first prototype. The main reasons being: hardware limitations of the available IC's, and the absence of the actual application
- we had gained essential architectural knowledge, on which the requirements and the design for the second prototype can be based.

step 5: requirements for the second prototype, derived from the experiences

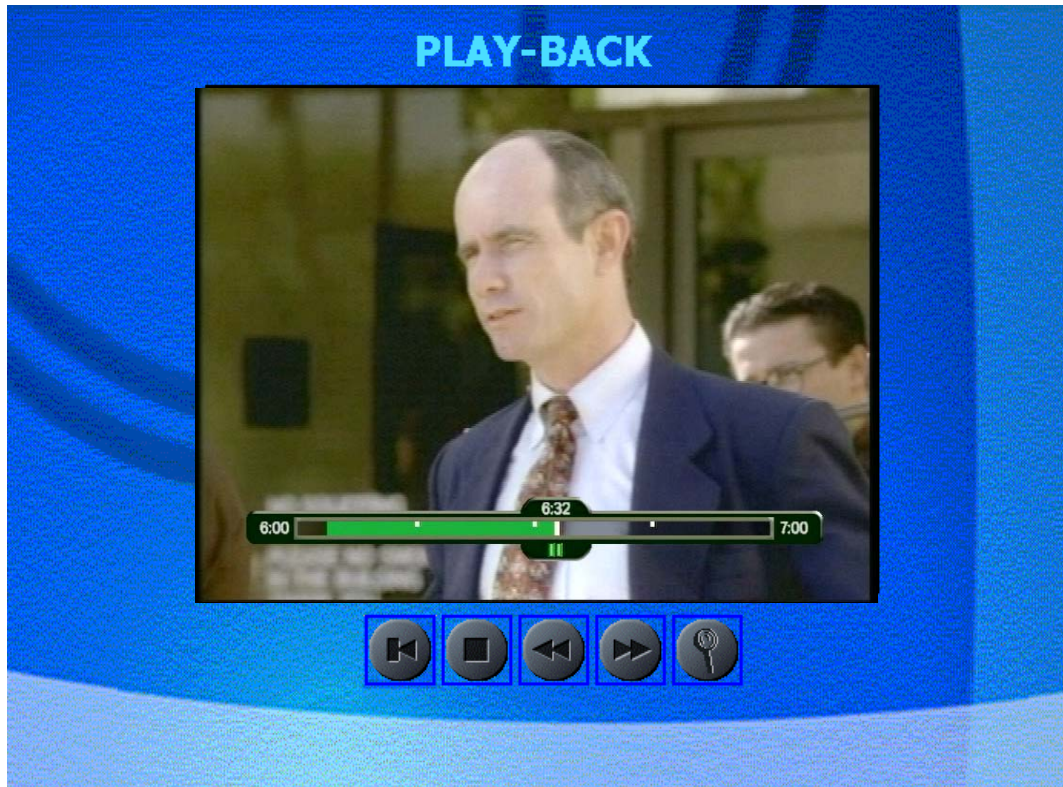
Apart from the limitations listed in step 4, the main objective of the 2nd demonstrator, as mentioned in deliverable D8, is to enable "digital time-shift".

Referring to the description of the list of requirements in deliverable D8, the following requirements will be added:

- [4 & 5], which have to do with scheduling multiple streams;
- [8 & 9], to allow jump forward or backwards in the audio/video stream.

As already mentioned, we encountered some roadblocks that prohibited us to implement all the features listed in document D2. To increase the added value of this research with respect to the required competencies in our business, we changed

our focus: for the second prototype we decided to concentrate also on user-interface-issues by providing buttons to navigate through the stream, and visual feedback of the current position in the Time-Shifted stream. Also the video would be shown in full screen mode or in a sub-window, which allows additional space on the screen for text, icons, buttons or other components to enhance the user-interface. A picture of the user interface items is given in Figure 3.



Buttons:

- Jump to beginning
- Stop / Play
- Jump backward
- Jump forward
- Full-screen

Figure 3 *The user interface for the time-shifted audio/video prototype.*

4.3. Application of the common reference model

As already mentioned, in this Beyond project, Philips Hasselt is concentrating on the technology needed to support adaptive consumer systems. Our primary focus was not on the adaptivity of the user interface itself but on the supporting technology. For this technology aspect, we focussed mainly on a robust real-time file-system, a sophisticated disk-scheduling algorithm, and integration with the winCE/MSTV s/w-stack.

Although the common reference model is a model on adaptivity of user interfaces, and not straight forward applicable on the time shift technology subject, the time shift project has gained important benefits of the insights by developing the common reference model, because of two reasons:

The first reason is that the time shift project has a relation with the “Adaptive System for Adaptive Content Retrieval Agents” project of Philips Research in Beyond, which concentrates on user interface adaptivity. The time shift functionality is developed for the advanced set top boxes that will support digital audio/video and offer advanced services. With “*an adaptive system for adaptive content retrieval agent*” Philips Research is developing part of these advanced services. The adaptive content filtering system will be embedded in e.g. an advanced set top box, offering the necessary supporting technology.

The second reason is that the time shift demonstrator, besides its main focus on supporting technology, also realises user interface functionality, by providing:

- buttons to navigate through the stream;
- visual feedback of the current position in the Time-Shifted stream;
- an option to view the video in a sub-window mode or in full screen mode.

For these user interface aspects, we focus primarily on ease-of-use, reliability and mechanical aspects.

4.4. Open issues

During the development of the two demonstrators, we discovered the following additional issues, which need further investigation:

- Most digital streams are scrambled, and need to be treated differently. This has a major impact on CA (conditional access), and on Trick-Play possibilities (fast-forward, jump, etc). The INCA-project is investigating the CA-issues further.
- Mounting a Hard Disk drive in a CE-chassis is not trivial, as there are conflicting mechanical requirements with respect to the areas of acoustics, thermal and performance.
- The ATA5-protocol is not really suited for real-time applications, such as audio or video recording/playback. Philips Hasselt and Philips Research are putting a lot of effort in the development of the new ATA6-standard, where the new key features for real-time-behaviour are: Command-Time-Limitation and Time-Sliced-Error-Recovery.
- The User-Interface of the current TIVO-box, our main competitor, is not optimal. One example is that the intelligent agents don't work well for households with multiple users. There is still quite some work to do in this area.
- During the project the market situation has changed: there is a need for dual analogue tuners. For boxes with dual analogue tuners, dual MPEG-encoders are needed. Where MPEG encoding for a single channel can be done in s/w, this is not the case for dual input.

4.5. Conclusions and recommendations

In this project we have studied the feasibility of time shift for advanced set-top box platforms. We followed an incremental way of development: a first prototype resulted in important knowledge on possibilities and limitations on the available IC's, on the architecture and on the absence of a top-level application. For the second prototype, we make use of this knowledge. We decided to focus also on user interface functionality. This will result in a second demonstrator showing the digital time shift amended with a user interface.

During the development of the two prototypes it became clear that much more effort is needed to investigate a number of issues: Conditional Access, hard disk mounting in CE products, a protocol for real-time hard disk applications (ATA6), a user interface for multiple users and dual MPEG encoders. Some of these subjects are already being investigated: development of ATA6 standard (Philips Hasselt and Philips Research), Conditional Access (Philips in the INCA project).

The second demonstrator will be presented to our customers. With our background knowledge gained in this project we will be able to interpret the customer evaluation results and to decide which issues need to be investigated first.

5. Development of the second prototype in the Vetronics domain

5.1. Introduction

In this section we describe the evolution of the first prototype of the Vetronics UI editor towards the second prototype of the editor.

BARCO and the LUC collaborate to realise ruggedised displays for use in several types of vehicles, for which the user interfaces can be defined in a flexible way. Vetronics (vehicle-electronics) are computer systems embedded in special-purpose vehicles, e.g. ships, trains, trams, air-planes or vehicles used in construction industry. Examples are global positioning systems in cars, trajectory guidance in busses and metro, systems allowing the monitoring of a boat in its environment etc.

On top of that hardware part we desire to have an 'intelligent' software layer having the most important feature that the User Interface is User Definable. The main aspect in the evolution of the first to the second prototype of the Vetronics UI editor is the architectural change to accommodate more flexibility.

BARCO collaborates with the LUC for the Vetronics software.

5.2. Lessons learned with the first prototype

The first prototype of the UI-Editor was meant to perform some research about the general structure of the Editor. This prototype gave us an impression of how to implement such kind of program and enabled us to learn about the do's and don'ts in this kind of application.

High-level and low-level requirements for the second prototype have been documented in deliverable D8, with special attention for specifications regarding functionality, general extensions, usability, adaptivity, multimodality, simulation and software architecture.

The first prototype met most of its functional requirements, except for some minor anomalies and bugs. However, we noticed a major lack in the application: Flexibility, by which we mean the ability to extend the application with new features, as such, following the hardware development.

The internal evaluation/presentation of the demonstrator was successful, but the same remark arose concerning future developments and the ease of extension.

5.3. Changed software architecture to improve flexibility

5.3.1. Off-line adaptivity

Strictly speaking, this document has to report about the "Application of the common reference model". In deliverable D3, a common reference model for adaptivity has been presented. The application of that reference model for Vetronics has been investigated. The conclusion was that, though future Vetronics systems might support on-line adaptivity, the current state-of-the art embedded systems lack the required power. Therefore, adaptivity in the Vetronics domain is currently restricted

to off-line adaptivity or flexibility in defining and reusing the UIs developed by means of the Vetronics UI editor. The rest of this section elaborates on the need for enhanced flexibility, and the architectural changes to realise more flexibility.

5.3.2. Need for enhanced flexibility

The need for enhanced flexibility was revealed by the evaluation of the first prototype of the Vetronics UI editor on the one hand, and by investigating additional functionality for the second prototype.

Though the first prototype of the UI editor offered the main and basic functionality to design a UI and download it to the target Vetronics system, the flexibility towards extensions of the UI editor was criticised. The current software architecture of the UI editor was sufficient to investigate the basic features. However, it was very difficult to extend the application with new features such as additional device drivers, to add components with functionality following from new hardware developments, and to offer enough flexibility for the tool user to add newly developed modules to an existing system.

On the other hand, it was envisioned to add more advanced simulation features in the second prototype, which could also be offered as separate modules to be added to the basic version of the UI editor.

Taking these two observations together, generally speaking, there was a need to evolve from an static application into a real component based application, which is open for new functionality. This can also be expressed as "Plug-in-capability".

5.3.3. Component-based architecture to support plug-in capability

The Vetronics UI Editor Tool needs to be flexible in multiple aspects. Extensions to the editor regarding hardware modifications or configuration changes must be supported, demanding an open approach. In addition, it must be possible to add functionality to the tool or some of its parts without compromising the whole design.

The Microsoft Component Object Model (COM) is a binary a platform-independent, distributed, object-oriented system for creating binary software components that can interact. It is a binary, architecture-neutral standard, and the foundation technology for Microsoft's OLE (compound documents), ActiveX (Internet-enabled components), as well as others.

COM specifies an object model and programming requirements that enable COM objects (also called COM components) to interact with other objects. These objects can be within a single process, in other processes, and can even be on remote machines. A COM object is a software object in which access to an object's data is achieved exclusively through a set of interfaces, which in turn are a set of functions or methods. Further, COM requires that the only way to gain access to the methods of an interface is through a pointer to the interface. Besides specifying the basic binary object standard, COM defines certain basic interfaces that provide functions common to all COM-based technologies, and it provides a small number of API functions that all components require.

By encompassing the functionality of the tool in components with a set of well-defined interfaces and a supporting framework, COM can easily match the need for flexibility. The component-based approach enables us to write additional components without compromising existing ones, and allows us to let them communicate with each other and the framework. The use of interfaces enables us to add functionality without compromising the one already present. As an added

bonus, the use of interfaces makes it possible to implement software patterns in a straightforward, transparent way, greatly simplifying design.

5.3.4. Relation to other work packages

The activities regarding the realisation of the Vetronics UI editor in BEYOND are mainly situated in work package 2 (adaptivity). The aforementioned evolution to a COM-based architecture is also an example of this kind of development activities. However, this architectural change has implications for activities conceptually belonging in other work packages (WP). These implications are briefly summarised here:

- WP1, *Multimodality*: We can implement every input device as a new plug-in by which we can cover all possible input sources (TouchScreen, Communication, ...).
- WP3, *Simulation*: The simulation environment will be incorporated into the UI editor as a plug-in. We based our HMI-code (Human-Machine Interface code) on a dedicated Virtual Machine (due to limited resources), which gives us the advantage that the simulation environment can interpret the generated (VM-) code. With regard to simulation, a design and a basic architecture have been realised, but the effective implementation is beyond the context of this BEYOND project.

5.4. Open issues

Though the evolution from the first to the second prototype of the Vetronics UI editor was done according to the planning, there remain several open issues in that prototype. These issues should be solved before further extension to a commercial product can be done.

- The integration of HW device drivers in Virtual Machine concept. This means the execution of native code from within the context of VM.
- Certification for Safety Critical applications.
- Performance of UI-Editor could be improved.
- Generated UIs must be thoroughly tested on the Vector System (target Vetronics system).
- The integration of UI-simulation in the project will be a big but important development.
- The main problems in the future are expected to be:
 - Resources and planning for commercialisation.
 - Certification for Safety Critical environments.
 - Continuous support for HW developments
- Future steps to be taken are in particular:
 - Extended Simulation
 - Speech/multimedia inputs
 - Significant performance improvement might need a hardware redesign.

5.5. Conclusions and recommendations

In this section we reported upon the evolution of the first prototype of the Vetronics UI editor towards the second prototype of the editor. The major change in this evolution was an architectural change in the UI editor. The motivation for the changed architecture, as well as some technical information (COM based plug-in architecture) have been described.

The second prototype of the Vetronics UI editor largely meets the original expectations as defined for this BEYOND project. However, it will need significantly more work to finalise it in the form of a commercial product. This is illustrated by a list of open issues and future activities.

The second prototype will be presented to the BARCO customers for evaluation, in order to assess the Vetronics UI editor and its relevance for the market. The commercialisation of the product will depend on the success of the presentation. Considering the previous 'limited' presentations already given, we are quite confident, so it is very likely that the program will be commercialised. This will mean a real breakthrough in the Vetronics market.

Benchmarking and feedback on exhibitions/conferences support this feeling. Currently, we are not aware of a similar product for the Vetronics domain.

6. General conclusions and recommendations

The objective of Workpackage 2 is to study the software technology needed in order to achieve an adaptive user interface. In this document the partners described their experiences in using the common reference model for adaptivity in creating adaptive user interfaces in the domains of avionics, home and vetronics. Some advanced prototypes are the result of this product, ranging from an intelligent flight deck to a multi-dimensional recommender for TV shows.

In general, the common reference model was successful in defining the 'common' aspects of developing adaptive user interface technology. Although the various prototypes covered the whole range from off-line adaptivity, or *flexibility*, to on-line adaptivity, involving a real-time tracking of user preferences and goals, the reference model could be used in all instances. Partners were complementary in their research focus: whereas the avionics research emphasised the adaptivity to context, the home research focused on adaptivity to user preferences.

All partners consider their second prototypes to be (very) advanced, and found it difficult to compare them with (in most cases not existing) products made or developments done at other groups and companies. This exemplifies that the goals defined for this Workpackage were quite ambitious, and the steps taken for most partners represented a true leap indeed for their software development.

The use of the multi-agent system paradigm as a framework to develop adaptive user interface software is considered to be a good choice. Strong points are the flexibility and expandability of the software, making it a suitable platform especially for purposes of integrating software. A weak point is in particular the rather slow real-time-ness. This is of course a drawback for user interfaces and remains an important area for further research.

Other main recommendations for the future are in particular related to the difficulty in tracking user behaviour (adapting to user preferences) and determining the user's goals (adapting to context). Much *fundamental* research, with less emphasis on generating prototypes but more on the fundamental cognitive processes of users in their interaction with systems, is needed to investigate new ways of inferring the intent of the user, and performance of existing algorithms still need to be improved to increase user acceptability.

7. References

Steentjes, A. & Mulder, M. (2000).: *Analysis of a Questionnaire addressing Free Flight's "Big Picture" Displays*. Report prepared for: Barcoviev (avionics division), Belgium. Published by: Delft University of Technology, Delft, 2000, 48 p.