



KnowledgeDatabase
MHP-KDB



D18: Guidelines on Migration

Considerations for markets migrating
from other APIs to MHP

Document / Version number:	D18 / v1.0
Date:	30.03.2006
Issued by:	The MHP Knowledge Project
Project Acronym / Reference:	MHP-KDB / 507442

Abstract:

This document provides assistance concerning questions which come up in markets which apply APIs different from MHP and want to migrate to MHP. It mainly gives hints to technical options how to broadcasting efforts during the migration phase can be minimised.

Keyword list:

DVB-MHP, proprietary APIs, Migration



Institut für Rundfunktechnik



Visit the MHP Knowledge Database: <http://www.mhpkdb.org/>

Deliverable D18	March 2006
Guidelines on Migration	

Document Information:

Document Type:	Deliverable
Document No.:	D18
Title:	Guidelines on Migration
Version No.:	1.0
Related to work package:	WP5
Type of the Deliverable:	Report
Dissemination level:	Public
Author:	Klaus Merkel
Due date:	Feb 2006
Delivery date (Ver. 0.9):	13 th February 2006 (Stable Draft)
Delivery date (Ver. 1.0):	31 st March 2006

Copyright notice

© 2006 Institut für Rundfunktechnik GmbH on behalf of The MHP Knowledge Project

This work may be reproduced and redistributed, in whole or in part, without alteration and without prior written permission, provided all copies contain the following statement:

© 2006 Institut für Rundfunktechnik GmbH on behalf of The MHP Knowledge Project. This work is reproduced and distributed with the permission of the Institut für Rundfunktechnik GmbH. No other use is permitted without the express prior written permission. For permission, contact info@mhpkdb.org.

This document is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

As we are interested to continuously improve the quality of our documents, we kindly ask you to report back any error you find in our documents or any improvement you are able to suggest. This can be done via writing comments into the database or by an email to feedback@mhpkdb.org.

Content

Content.....	3
1 Introduction.....	4
2 Basic technical architecture in API markets	5
3 General migration considerations and options	7
3.1 Converting the decoder population.....	7
3.2 Providing the applications.....	7
3.2.1 Sharing resources - saving data rate.....	8
3.2.2 Avoiding double application authoring effort.....	9
4 Technical aspects of shared resources.....	12
4.1 Loading / interpretation performance.....	12
4.2 Content formats	12
4.3 Transport protocols.....	13
4.4 Data compression.....	13
5 Migration: example market.....	14

Deliverable D18	March 2006
Guidelines on Migration	

1 Introduction

This document provides assistance concerning questions which usually come up in markets or single platforms which apply APIs different from MHP and want to "switch" to MHP.

There may be various reasons for such a switch. Unifying APIs in a certain region in order to achieve a more open and competitive development or implementing MHP as a more flexible and future proof API could be such a reason.

The investments made in the existing services and infrastructure, however, will in most cases be a serious obstacle to a simple API "switching", consequently, a carefully planned "migration" strategy is required.

This migration process is related to economical and regulatory questions and also to the detailed market structure and the specific situation of the respective market or in a specific country. But it also implies a number of technical aspects and they are focus of this document.

This document aims at encouraging markets using proprietary APIs to go for MHP by giving technical hints and examples how a migration can be achieved without too much effort.

In this context, neither the introduction of MHP in a non API market is considered as "migration", nor is it the introduction of a new MHP version in an existing MHP market. The latter aspect is covered in chapter 7.5.2 of "The MHP Guide" (available at <http://www.mhpkdb.org>).

2 Basic technical architecture in API markets

This chapter provides some basic architectural schema which is common for all API systems. Specifically for the case of MHP, a more detailed architectural description can be found in chapter 5 of "The MHP Guide" (available at <http://www.mhpkdb.org>).

Seen from a very high degree of abstraction, applying an API to a broadcast chain looks like depicted in **Figure1**.

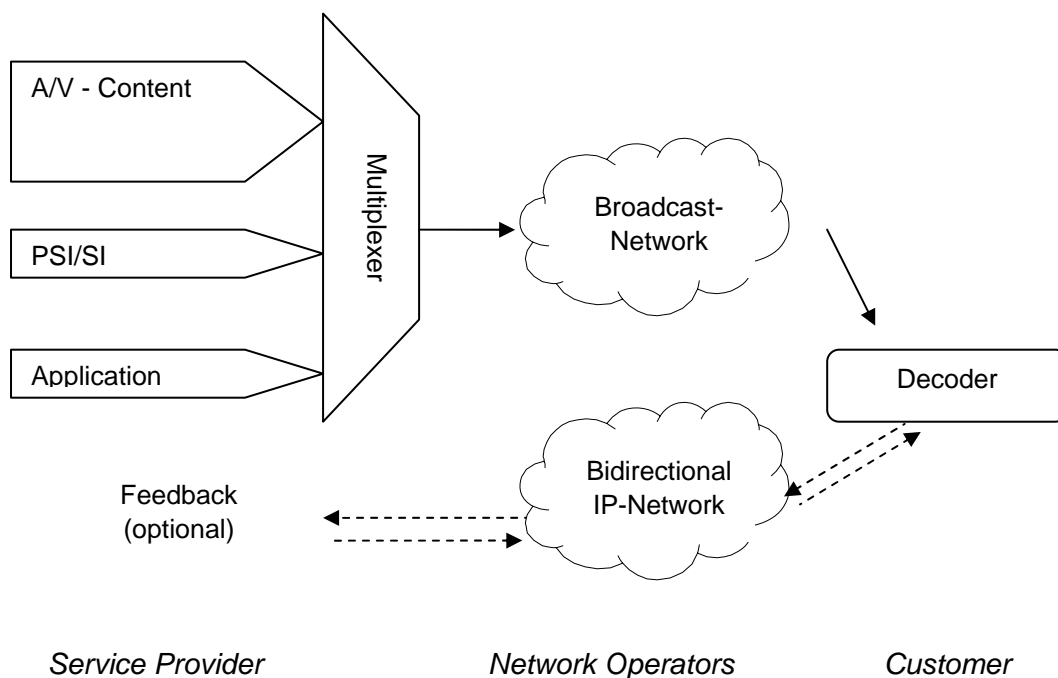


Figure 1: Applications in the broadcast chain

In the case of creating an interactive version of a regular TV service, the service provider adds the interactive application to the standard audio and video (A/V) components of the offered service. This is done in the service multiplexer and this adds also some more data rate to the complete data stream which has to be carried over the network. As in broadcast networks data rate is an expensive resource, the required data rate for an application is an important factor in all business models related to interactive television.

Another component in the broadcast stream which can also be of some relevance in the migration context is the "service information" (SI) and the "program specific information" (PSI). The PSI/SI contain information concerning the actual service and its schedule. The PSI/SI can contain only a few data but can also carry a lot of programming information and thus consume an according amount of data rate. The PSI/SI data can be read and displayed independently from any API by all standard DVB decoders.

Looking closer at the application component, it can be seen, that most applications can be – at least at the functional level – separated into different sub components.

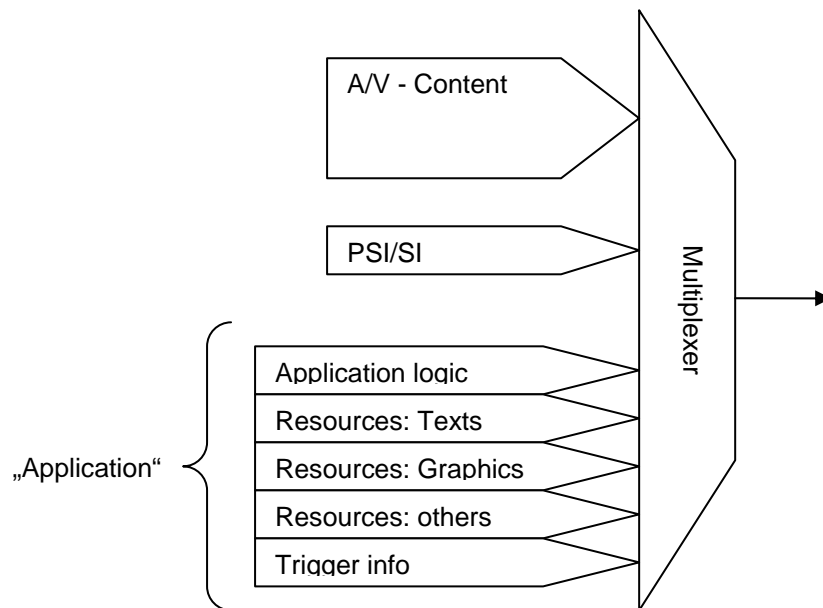


Figure 2: Applications in the broadcast chain

As shown in **Figure 2**, an application generally consists of

- Parts of code for the execution engine of the decoder: this part of the application controls all the basic functions and interactions and provides the overall “logic” of the application.
- Resources like text elements: those texts are included to inform the user; they can be static (e.g. to provide general information about the application) or dynamic (e.g. the news texts in the news section of an application).
- Resources like graphics: those graphics can be full screen images (e.g. as background images), smaller images (e.g. as snapshots related to news texts) or even small logos, icons or interactive click-buttons
- Other resources to provide ordering info, links, layout support, etc.
- Trigger info which triggers a new state in an application (e.g. a new question in an interactive quiz).

Structures like this can be found across various application types and various API systems; their implications for the migration process are discussed below.

Deliverable D18	March 2006
Guidelines on Migration	

3 General migration considerations and options

Migrating a market from a proprietary API to MHP does normally mean that at the beginning of the migration process all applications in the broadcast networks are implemented and broadcast in the proprietary API X, and all decoders in that market are only equipped with this API X. At the end of a successful migration process, both applications and decoders will be converted to MHP.

It is not acceptable in most cases to introduce MHP decoders which additionally support API x. This would avoid any simulcast at the beginning, but would cost technical as well as licensing overhead and not lead to a standardised and open environment within a limited time span.

3.1 Converting the decoder population

MHP is based on JAVA which is a modern, powerful and flexible programming language, but consumes more hardware resources than the usual proprietary APIs. This means, that – unless explicitly planned beforehand in the onset of the service – a simple software update of the decoder will probably not be sufficient to upgrade all decoders in the market from API X to MHP.

As the decoder population represents a major investment, a re-investment is, on the other hand, a challenge:

In *vertical markets*, the operator normally owns the decoders, but the high investment in a full population of new decoders will be deterring in most cases. The operator would prefer to exchange the decoders over a certain time span, e.g. two years, to make use of natural fluctuation effects and some decoder replacement which would be required anyhow.

In *horizontal markets*, there is no single entity to control the decoders in the households. The investment has been made by individuals who will be reluctant to reinvest ad hoc. But provided that more attractive services are offered in MHP and taking into account that the normal lifetime of an API decoder is generally much less than 10 years, it can be assumed that within some few years most of households will have converted to MHP.

So a conversion of the decoder population to MHP is possible, but it will take some time to happen.

3.2 Providing the applications

The consequence of the inertia in converting the decoders is that, for some time, the relevant applications have to be provided in both APIs in order not to disappoint or loose audience.

At first glance, providing applications in two API versions does imply twice the application data rate in the broadcast channel which would increase the expenses in this respect or the additional data-rate would be hardly available at all. Furthermore, all applications would be to be developed twice, once for the API X and once for MHP. Depending on the number of new applications during the migration phase, this would also imply a duplication of effort and costs.

Both aspects are dealt with in the following sections.

3.2.1 Sharing resources - saving data rate

For interactive TV programmes, it is rather obvious that the audio and video component can be shared among the two interactive versions of that programme as shown in **Figure 3**:

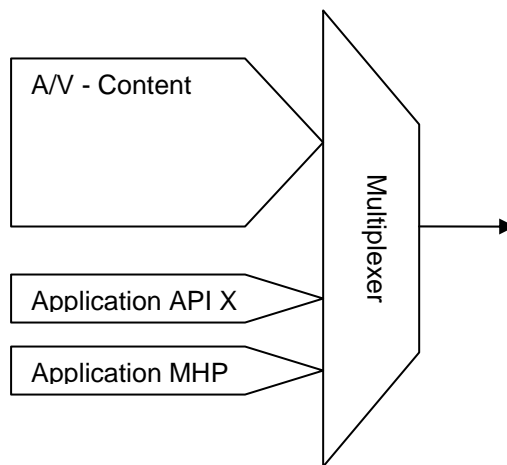


Figure 3: Two APIs serving one A/V-service

Assuming that the data-rate for audio and video is 4 Mbit/s and the data rate for the application is 1 Mbit/s, the data-rate for the interactive service is 5 Mbit/s. But providing this service in two API versions does not require $5+5=10$ Mbit/s but only $4+1+1=6$ Mbit/s.

Sharing resources however can be applied to an even higher extent. Also some of the components within the applications as they have been introduced in **Figure 2** can be shared in such a way.

Luckily, in the majority of applications, the percentage of data-rate which is needed for components suitable for sharing is rather high and can reach 90 % or more.

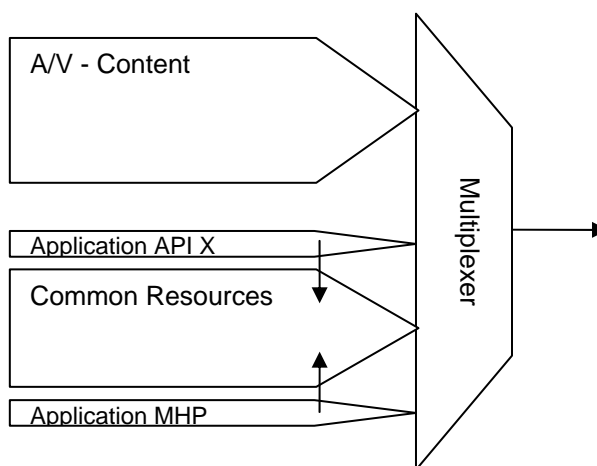


Figure 4: Common resources used by two different applications

This basic concept allows realizing a migration process as indicated in **FIGURE 5**.

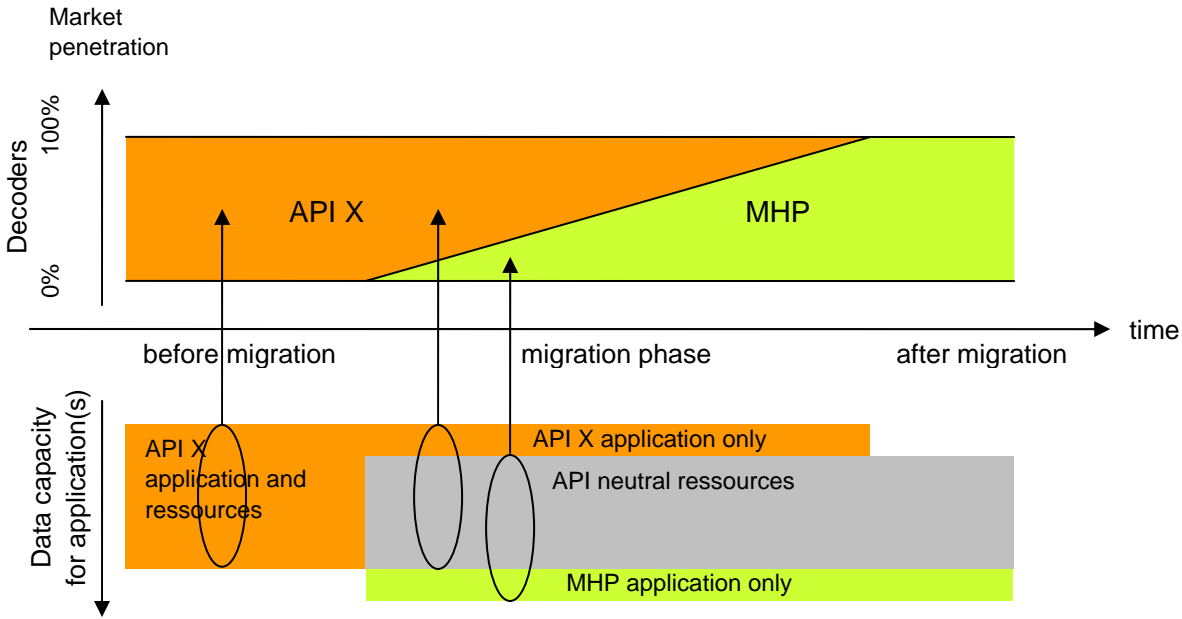


Figure 5: Migration phases

Before the migration process begins, an application consumes a certain data capacity. At the beginning of the migration phase, at latest, the application has to be designed in a way to allow reading the resources by the possibly redesigned application in API X, but also by an MHP application. This MHP application is then added at the beginning of the migration phase. As the resources usually make the majority of the required data capacity, just a relatively small additional data rate is necessary to provide this additional MHP application whilst granting full functionality of the interactive service for decoders of both APIs.

After the migration phase, the application serving the “old” decoders equipped with API X can be switched off.

Some technical detailed aspects of this procedure are dealt with in chapter 4.

3.2.2 Avoiding to double the application authoring effort

If there is more than one application or new applications are to be launched during the migration phase, the effort for developing all applications for two APIs is certainly higher than doing so only for one.

Even if the majority of the work related to new applications is not consumed by the application programming itself, but by the development and proof of the concept, provision of graphics and data resources and integration into the production workflow and so on, it is desirable to have as little overhead as possible.

There are two approaches to deal with this aspect: the browser approach and the approach of using a Portable Content Format (PCF).

3.2.2.1 The browser approach

This variant can be used if the application has a limited and fixed number of basic functions and is mainly page-oriented like an enhanced teletext or other simple information services.

In this case, the content of the application can be completely described using a mark-up language like HTML or XML. The corresponding HTML or XML data files can then be broadcast as the resource component of the interactive application.

They are interpreted by specifically optimised browser applications which are implemented for each API and which are not too bandwidth consuming.

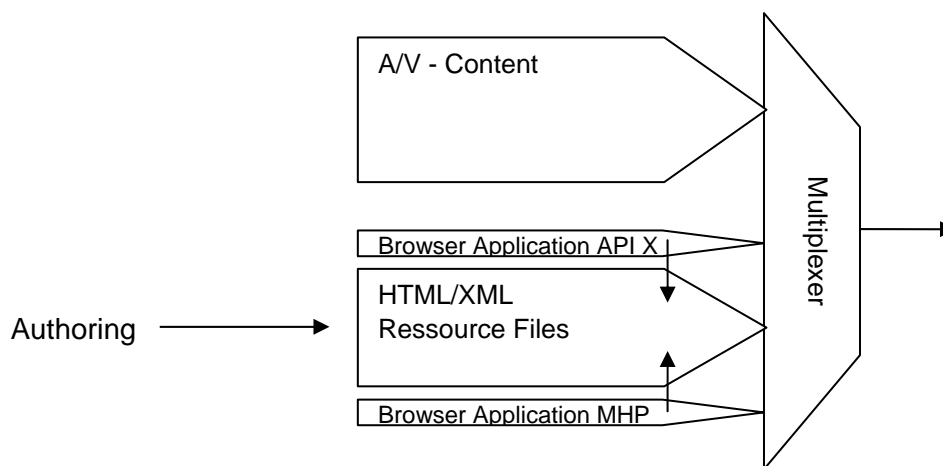


Figure 6: API specific browsers for common content

The authoring for the interactive service is done only once and both browser applications which access the resource files (in the API decoder, not in the play out!) remain unchanged.

As long as there is no complex functionality, this is a very effective way to use standard design tools to create a number of very individually looking interactive services.

3.2.2.2 Using the Portable Content Format

At the time of publishing this document, the technical specification work within DVB for a "Portable Content Format" (PCF) has been finished, but the approval by ETSI is not yet accomplished.

The main aim of the PCF is not specifically to ease migration, but to generally provide a tool to support content creation for more than one API system. Besides for migration, this is of great interest when interactive content has to be fed into two or more networks using different APIs. For this aspect, which is illustrated in **Figure 7**, the aspect of excess data-rate is not relevant as the full services have to be broadcast in all relevant networks anyhow and there will probably be only one API per network.

Whereas the PCF interfacing format itself is specified by DVB, the specification does not cover the "converters" into the specific API systems. Their implementation is vendor or even network specific.

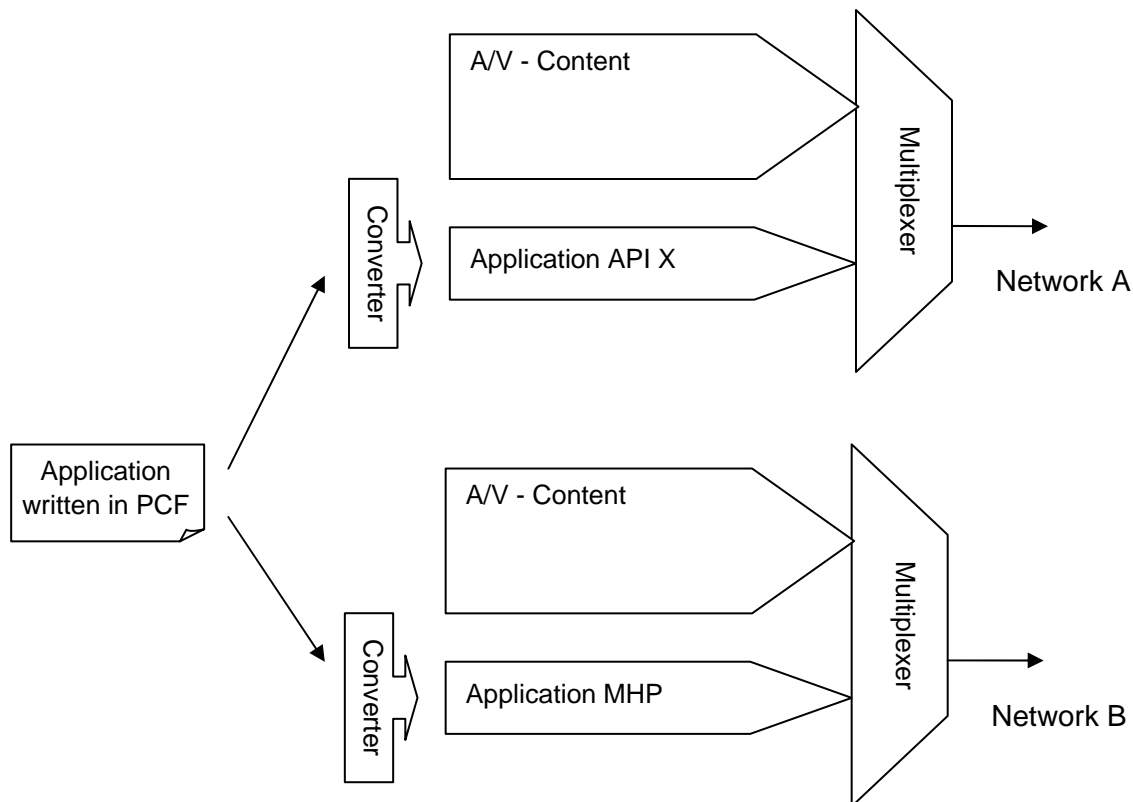


Figure 7: Using PCF for different networks

As shown in **Figure 7**, the request for potential synergies in terms of resources is of no relevance when feeding different networks with different APIs. PCF converters for both APIs can be optimized independently in each case.

In the migration context, however, a PCF converter has to be optimized for all relevant APIs in order not to save only authoring effort but also data-rate (i.e. transmission capacity) in the network. **Figure 8** depicts the situation where PCF is used to ease authoring during the migration phase.

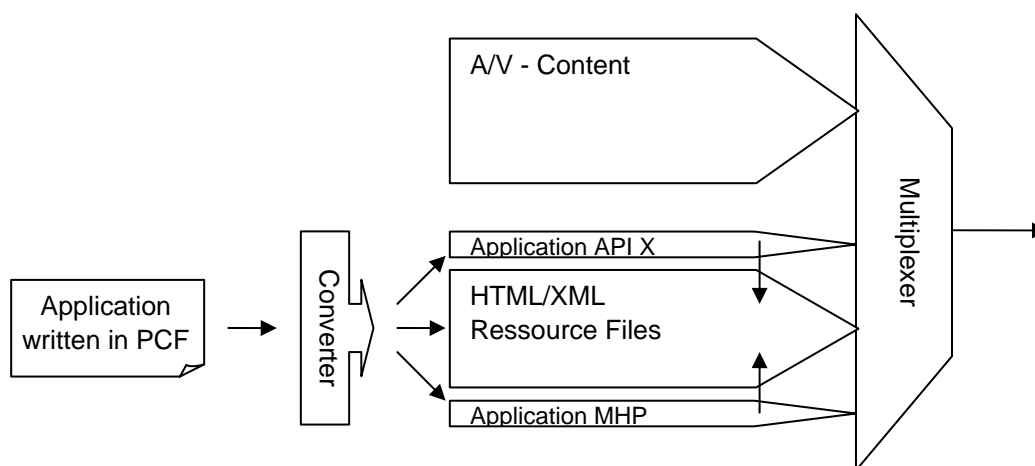


Figure 8: Using PCF for API simulcast

Deliverable D18	March 2006
Guidelines on Migration	

4 Technical aspects of shared resources

When sharing resources between applications of different APIs, several technical aspects have to be taken into consideration. Those aspects have generally a different impact depending on the type of resource (or component as it was introduced in chapter 2).

As a consequence, no simple technical concept can be derived to realize the optimum technical migration strategy. As conflicting aspects occur, an individual balancing is required for each specific application in order to achieve the advantages aimed at.

The relevant components to look at are:

- Texts
- Graphics
- PSI/SI
- Trigger info

The applications themselves are by definition not suited for sharing and can be ignored here. Other resources not fitting in the scheme above have to be treated individually in each specific case.

PSI/SI is included even if they are basically independent from any API. As they are used in many interactive information banners or full EPGs, they can provide a valuable data source for applications.

4.1 Loading / interpretation performance

As a first general rule, care has to be taken that making resources compatible to applications of both systems does not slow down the times for loading and interpreting them considerably. As soon as the performance of loading/interpreting goes down to 50% of the former performance when using a format optimised for one API, it becomes useless to share the resources in order to save data-rate: In this case the application could be generated independently for both APIs and simply be simulcast at a data-rate of 50 % of single-case data-rate.

4.2 Content formats

The preferred text format for MHP applications is UTF-8 [cf. MHP Specification 1.0.3 (ETSI TS 101 812), section 7.1.5], but also other standard text formats can be imported.

For simple graphic elements the PNG format is mandatorily supported by all MHP terminals. The GIF format is supported only optionally.

For full size background images, the content format is usually that of an MPEG-2 still frame which can be decoded very fast and effectively by the hardware video decoder chip. Thus Background images can be used in various API systems.

PSI/SI information is standardised anyway and thus API- independent. PSI/SI can be read easily by many API systems.

Deliverable D18	March 2006
Guidelines on Migration	

4.3 Transport protocols

MHP uses DSM-CC object carousel as standard transmission protocol. The MHP decoders are equipped with advanced routines to load all data files from this carousel effectively. So it would be preferable to use DSM-CC object carousel also for API X. If not available at the beginning, the option to upgrade API X to DSM-CC should be checked.

If DSM-CC object carousel is definitely not available, then the underlying protocol level, which is MPEG-2 sections, can probably be used.

For MHP decoders DSM-CC is mandatory only for the application itself – which is not a problem as it is API specific anyway. Other data resources can also be read by an MPEG-2 section filtering API which is a mandatory part of all MHP decoders.

As MPEG-2 sections are the most common transport basis for all types of non real-time streaming data, it will most likely be also used for API X and can thus be used as a common transport layer.

Also PSI/SI data is transmitted via MPEG-2 sections. Specifically the EIT can be the most important data source for EPGs and it can reach rather high data rates of 2 Mbit/s or more. As APIs are usually equipped with EIT reading interfaces, sharing this component is fairly easy and very effective.

Short messages useful for triggering events are sent in MHP via DSM-CC stream events preferably. Alternatively they can be transported via MPEG sections as well and thus potentially also be used by other APIs. On the other hand, in terms of saving transmission capacity, it causes not much overhead to simulcast such messages, if more convenient, as they will be rather compact in most cases.

4.4 Data compression

The MHP specification supports the compression of data modules. For transmission, the ZLIB Compressed Data Format Specification version 3.3 is foreseen (IETF RFC 1950 / May 1996). The decoder has to support the DEFLATE Compressed Data Format Specification version 1.3 (IETF RFC 1951 / May 1996) [cf. MHP Specification 1.0.3 (ETSI TS 101 812), Section B.2.9].

The compression of modules is generally a topic to be considered: as the decompression costs processing power on the MHP terminal and can thus slow down performance it should not be used for very small modules or for modules which contain already highly compressed (e.g. MPEG2 stills) or quasi-random data. On the other hand, compression can be effective for bitmaps or longer texts and thus save transmission capacity or allow faster loading times.

If compression is considered in the context of sharing resources, it is important that the same compression algorithm is implemented in both API systems. As for the protocol issue, it may be an option to implement missing compression algorithms either on the MHP or on the API X decoder in a native way. This could also be achieved via software update of one of the decoder types. A software implementation via a broadcast application would be possible in principle, but will be detrimental to the performance.

Deliverable D18	March 2006
Guidelines on Migration	

5 Migration: example market

In Germany, an API migration has successfully been performed from the OpenTV system towards MHP.

OpenTV had been introduced in 1997/98 due to the absence of a standardised API system. But a later migration to MHP which had its development started by 1997 was aimed at from the very beginning of OpenTV broadcasts.

After first MHP demonstrations at the German IFA in 1999, regular MHP services were taken up during 2002 by some broadcasters when the MHP implementation had reached a sufficient degree of maturity.

The broadcaster providing the majority of the interactive applications has been and still is ARD. For the migration from OpenTV to MHP a simulcast phase was foreseen to support both OpenTV and MHP decoders for some time.

At the time of the OpenTV start, all applications had individually been optimised only for this system. With the advent of MHP, applications were redesigned in the way described below to support the migration. Meanwhile, as OpenTV transmissions have been terminated completely, further MHP development can be done independently of any migration considerations.

The ARD applications have been treated as follows during the migration phase:

ARD portal / EPG

The portal and EPG application itself is relatively small and consumes only a few hundred Kbit/s. It uses a lot of SI/EIT data to display all events covering some 18 TV programmes and several days of preview. The data rate for these EITs is in the range of 2 Mbit/s and can be used for MHP as well, using the SI API. For MHP terminals a completely independent portal/EPG application was developed consuming the same little transmission capacity as the OpenTV application but providing the same full EPG service.

Superteletext

The superteletext application consuming some 3 Mbit/s consists of lots of background stills and also some audio tracks. Since the beginning, it was based on HTML content creation. For the migration phase, it was redesigned to an improved browser concept based on a XML content format. Relatively small OpenTV and MHP browsers were broadcast, whilst the bulk of XML content and also the audio tracks was shared between both systems as discussed in **Figure 4**.

Game shows

Game shows including interactive add-ons had been performed from time to time during the OpenTV phase. As the audience had not got used to them so much, they were not supported in both API versions, but the MHP variant simply replaced the OpenTV one. Thus any overhead was avoided.

The migration itself was achieved successfully. Applications initially using a total of 6 Mbit/s for OpenTV only could be supported during the migration phase for both API versions while *not* requiring 12 Mbit/s of data rate, but only an overhead of some 10 % of data rate.