Deliverable 3.4 LinkedTV Interface and Presentation Engine version 1

Pieter van Leeuwen

Jechiam Gural

Rutger Rozendal

13th October, 2012

Work Package 3: LinkedTV platform

LinkedTV

Television Linked To The Web

Integrated Project (IP)

FP7-ICT-2011-7. Information and Communication Technologies

Grant Agreement Number 287911

Dissemination level[1]	PU
Contractual date of delivery	*30th September 2012*
Actual date of delivery	*13th October 2012*
Deliverable number	D3.4
Deliverable name	*LinkedTV Interface and Presentation Engine version 1*
File	*LinkedTV_D3.4.*doc
Nature	*Report*
Status & version	V1.0
Number of pages	*23*
WP contributing to the deliverable	*WP 3*
Task responsible	*Noterik*
Other contributors	*Pieter van Leeuwen, Jechiam Gural, Rutger Rozendal Noterik*
Author(s)	*David Ammeraal, Noterik*
Reviewer	*Katarina Stanoevska-Slabeva University of St.Gallen*
EC Project Officer	*Jorge Santos*
Keywords	
Abstract (for dissemination)	

[1] • PU = Public
• PP = Restricted to other programme participants (including the Commission Services)
• RE = Restricted to a group specified by the consortium (including the Commission Services)
• CO = Confidential, only for members of the consortium (including the Commission Services))

Table of contents

List of Figures

1 Introduction

The functional requirements of the LinkedTV presentation engine are highly depending on the user scenarios and amount of personalisation inside this scenarios. In this deliverable we will describe how the mock-ups that have been constructed by partners within WP6 are translated to technical requirements for the design of the presentation engine.

In order to illustrate the 'flexibility' of the used XML schema and the play-out components of the Noterik player we will give a brief introduction into the technical foundation of the Presentation Engine. A more deliberate technical description can be found in Deliverable 5.2 '*LinkedTV front-end: video player and MediaCanvas API*'.

1.1 Related LinkedTV deliverables

The design requires the consideration and input of almost all LinkedTV packages. However, D3.4 strongly relates to the following deliverables in particular:

- *D3.2 Specification of Presentation Interfaces for the Three Scenarios*
- *D3.3 LinkedTV User Interfaces Sketch*
- *D5.1 LinkedTV Platform and Architecture*
- *D5.2 Video Player Media Canvas API*

1.2 History of the document

Table 1: History of the document

Date	Version	Name	Comment
2012/7/5	V0.01	Gural, Noterik	Initial document structure
2012/7/16	V0.1	van Leeuwen, Noterik	Initial structure for section 2 and 3
2012/7/30	V0.2	van Leeuwen, Noterik	Initial structure for section 4
2012/8/5	V0.3	van Leeuwen, Noterik	Refinement section 2 and 4
2012/8/27	V0.4	van Leeuwen, Noterik	Updated document with input from D3.3
2012/9/4	V0.5	van Leeuwen, Noterik	Added structure for section 5
2012/9/6	V0.6	Van Leeuwen, Noterik	Refinement of section 5

Date	Version	Name	Comment
2012/9/14	V0.7	Van Leeuwen, Noterik	Updated introduction
2012/9/19	V0.8	Ammeraal, Noterik	Revised 2.1, Revised 2.2.1 Addition made to 3.1 concerning Gestures, Revised 4.4, Updated figure 12, Added figure 13
2012/10/08	V0.8	Stanoevska-Slabeva	Review und suggestions directly in the text
2012/10/09	V0.9	Rutger Rozendal, Noterik	Inserted a new introduction, clear references to 5.2, overview of the way the Presentation Engine is embedded into the system, description of the Display Database and Presentation XML.

1.3 Abbreviations and Acronyms

Table 2: Abbreviations

Abbreviation	Explanation
AJAX	Asynchronous Javascript and XML
API	Application Programming Interface
CSS	Cascading Style Sheet
DLNA	Digital Living Network Alliance
HbbTV	Hybrid broadcast broadband TV
JSON	JavaScript Object Notation
QR code	Quick Response code
REST	Representational State Transfer
SOA	Service Oriented Architecture
UI	User Interface
XML	Extensible Markup Language

2 Functional Requirements

This section describes the functional requirements for the Interface and Presentation Engine. These functional requirements are present in two different categories:

- The general requirements that are defined by the project.
- The requirements from the different scenarios.

Based on these requirements we will motivate the design and structure of the Interface and Presentation Engine.

2.1 Project requirements

The general project requirements are derived from the project description. The project aims to support a broad range of devices including televisions (using HbbTV), computers and mobile devices. In order to support this range of devices, HTML5 will have to be used. This because it is available on modern devices like computers and mobile devices, offers support for audio & video and is proposed to be supported in HbbTV version 2. This makes HTML5, even though still in the standardization process, a suitable language for developing the client part of the Interface and Presentation Engine.

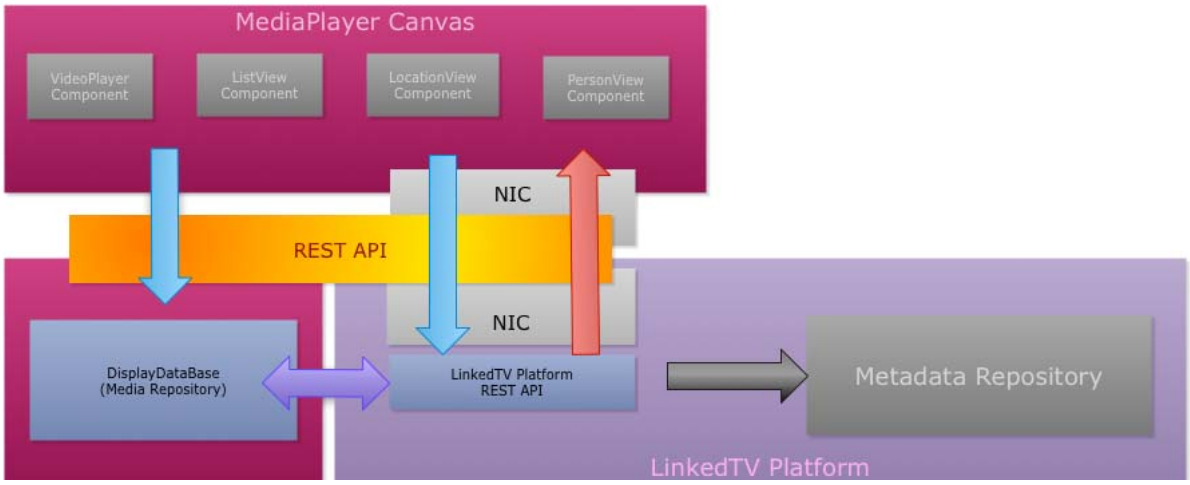

Figure 1: Overview Media player LinkedTV

The above picture shows the architecture of the media player of the LinkedTV platform. The purple part shows the MediaPlayer Canvas, which will be build in HTML5. The Presentation Engine consist out of the Media Player Canvas and the XML containing the information about a specific presentation, which is stored in the Display Database.

2.2 Scenario Requirements

Partners within the LinkedTV project have created several scenarios that describe how users will interact with the system, what they expect from the system using different kind of technology. These are the requirements derived from the scenarios mock-ups in deliverable 3.3. The partners responsible for the scenarios within the LinkedTV project are the Netherlands Institute for Sound & Vision and the German broadcaster RBB. From both we have taken their mock-ups for a television screen and second screen application. Within these two application views the partners worked out two different options on how a user could use the applications to get more data from the video.

2.2.1 Mock-up Sound & Vision - Television screen

These mock-ups operate in the context of a single television screen for displaying both the video and the additional information relevant to the video. This affects the user interaction in such a way that it should be optimized for usage with a remote control.

2.2.1.1 List view mock-ups

In the list view mock-ups we can identify three different parts in the interface. First the display of the video, either covering the whole screen or part of the screen. Secondly a list view bar is displayed providing a list of items about topics from the current scene that have extra information available. The user must be able to browse these topics if there is more then one available and select a topic. The third main aspect of this interface is the additional information resources window. This window holds a list about the available resources from the currently selected topic from the bottom bar. The user must be able to browse through all additional resources and the content of that resource will be displayed aside of the list. As the additional information resources can hold detailed images or large texts the component must be able to be displayed at a larger size.

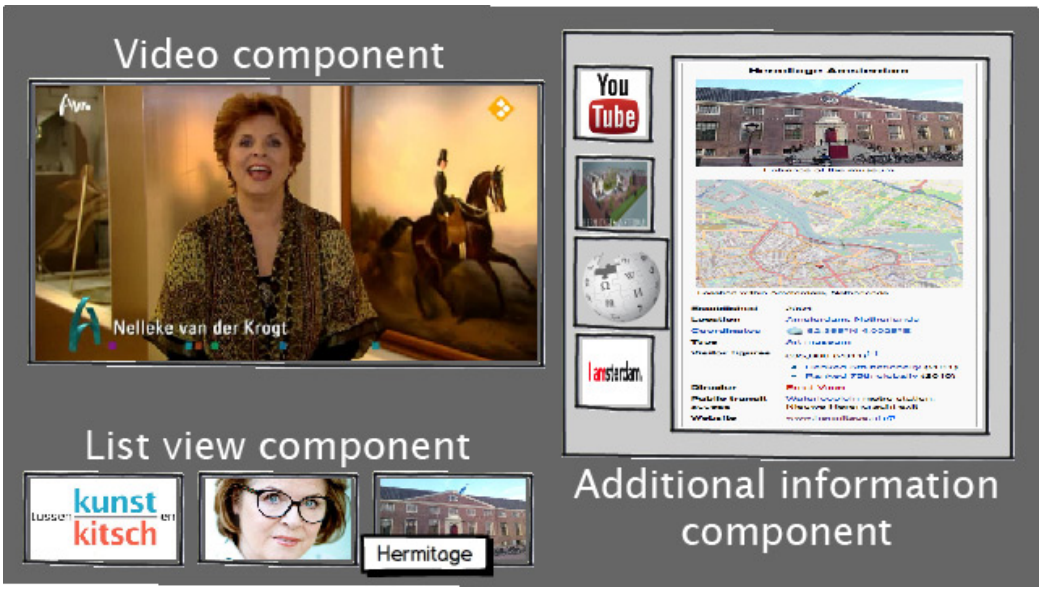

Figure 2: Television screen mock-up – list view

2.2.1.2 Timeline view mock-ups

The timeline view mock-ups differ from the list view mock-ups by holding a timeline instead of a list of topics in the bottom. The timeline component shows the available topics that hold additional information in the timeline on their position in the video. When a user selects an additional information resource the timeline shows a sidetrack with the resource under the current topic. The additional information is presented in a slightly different way. Instead of a side list of all the additional resources on a topic this data is now presented above the selected resource. Since this is just a minor difference in how the data is represented the component stays the same as in the list view mock-up.

Figure 3: Television screen mock-up – timeline view

2.2.2 Scenario Sound & Vision - second screen

The second screen scenario mock-ups distinguish from the television screen mock-ups by presenting all additional data on a secondary device instead of on the same device or window as the video. This separation results in the need for a central component that allows communication between the different devices in order to maintain a sync between both the video and the data displayed on the second screen.

2.2.2.1 Timeline view mock-ups

The components for the second screen timeline view mock-ups are identical to the components presented in the television screen timeline view mock-ups, with the major change that the data components have to be displayed on the secondary device while the video will be displayed on the television screen.

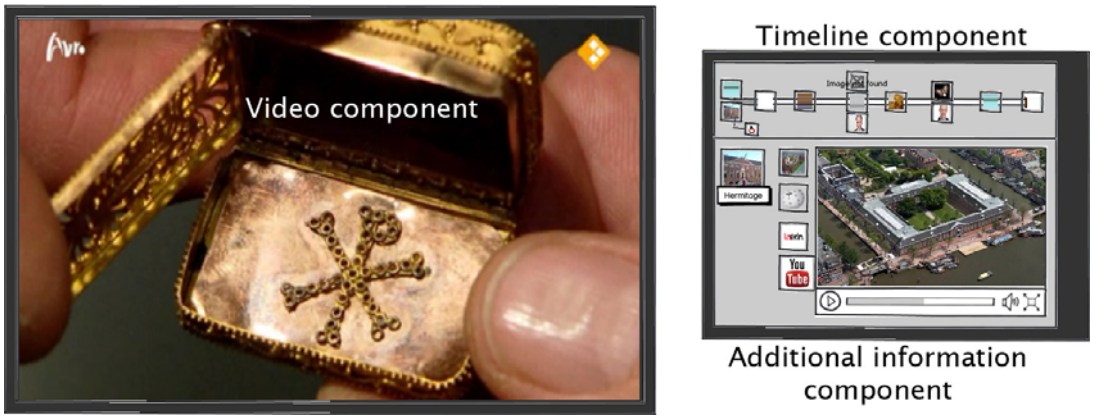

Figure 4: Second screen mock-up – timeline view

2.2.2.2 Table view mock-ups

The table view mock-ups introduce a new table component that holds all topics and additional resources together. Every row in the table holds a topic with the first column representing the topic. All the other columns represent the available additional resources. If the user selects an additional information the data is represented next to the first columns.

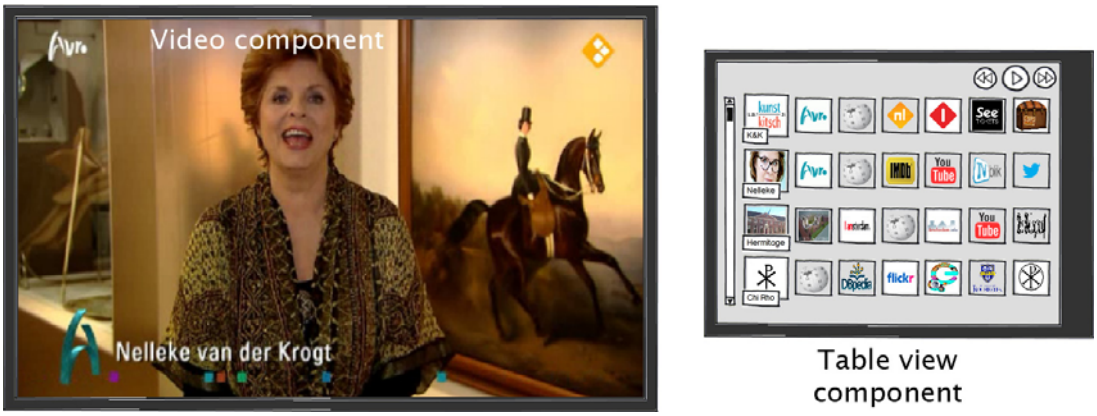

Figure 5: Second screen mock-up – table view

Figure 6: Second screen mock-up – table view

2.3 Scenario RBB television screen

The television screen mock-ups from RBB have the same technical target as the mock-ups from Sound & Vision in section 2.2.1. However the target audience is different. Where Sound & Vision aims for a casual user that might be interested in more related info and can request this on his demand RBB targets users that want more related information on default.

2.3.1 List view mock-ups

RBB took a different approach of presenting the additional information in a list view compared to the Sound & Vision mock-ups. By default the video does not show in fullscreen but shows already a timeline below the video that indicates when additional information is available. On the right side of the video the available additional information resources on the current topic are visible.

Figure 7: Television screen mock-up – list view

Once new topics are shown in the video, the newer topics become available next to the older topics, allowing the user to view all additional information resources of the entire show once an item played. When the user clicks an additional information resource this information will displayed under the list.

Figure 8: Television screen mock-up - list view

2.3.2 Timeline and circle view mock-ups

The timeline and circle view features the same timeline as in the television view. However on the right side the circle view gives a centralized topic with around it all the available additional information resources around it. Once a resource is requested it will shows up under the circle view and can be put to fullscreen.

Figure 9: Television screen mock-up – timeline and circle view

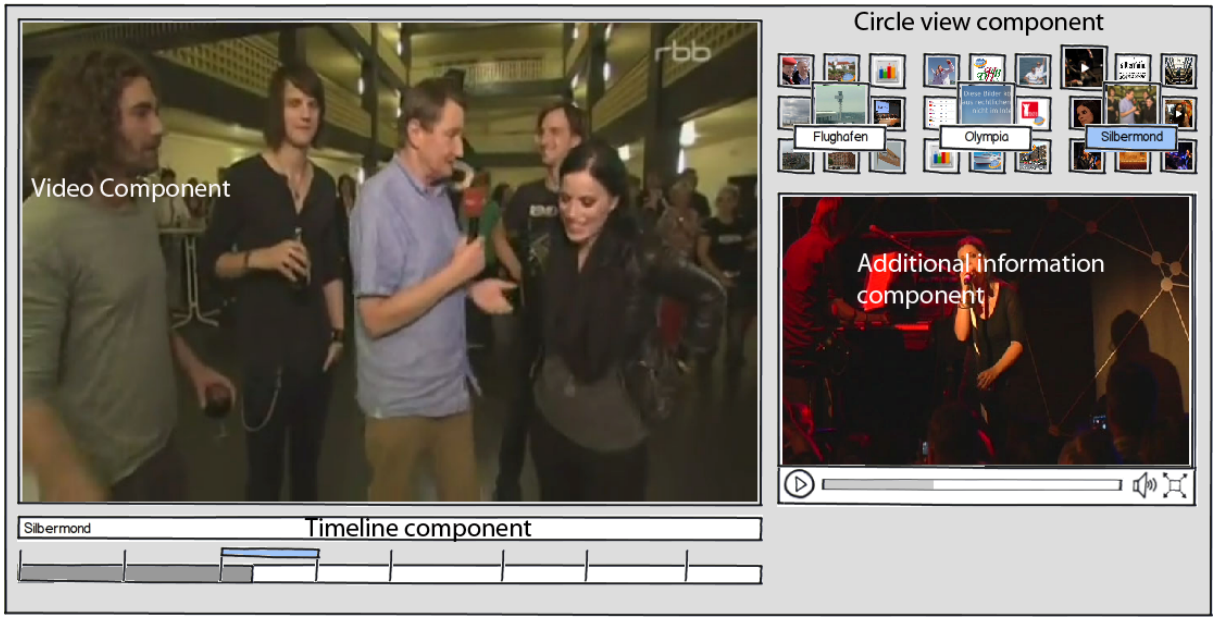

Figure 10: Television screen mock-up – timeline and circle view

2.4 Scenario RBB second screen

The second screen scenarios from RBB have, just like the second screen scenarios from Sound & Vision, all the data represented on the second screen leaving the television screen only for displaying the video.

2.4.1 Timeline and list view mock-ups

The timeline and list view mock-ups feature the same elements as in the television equivalent. This includes switching related videos from being displayed on the second screen to be displayed on the television screen.

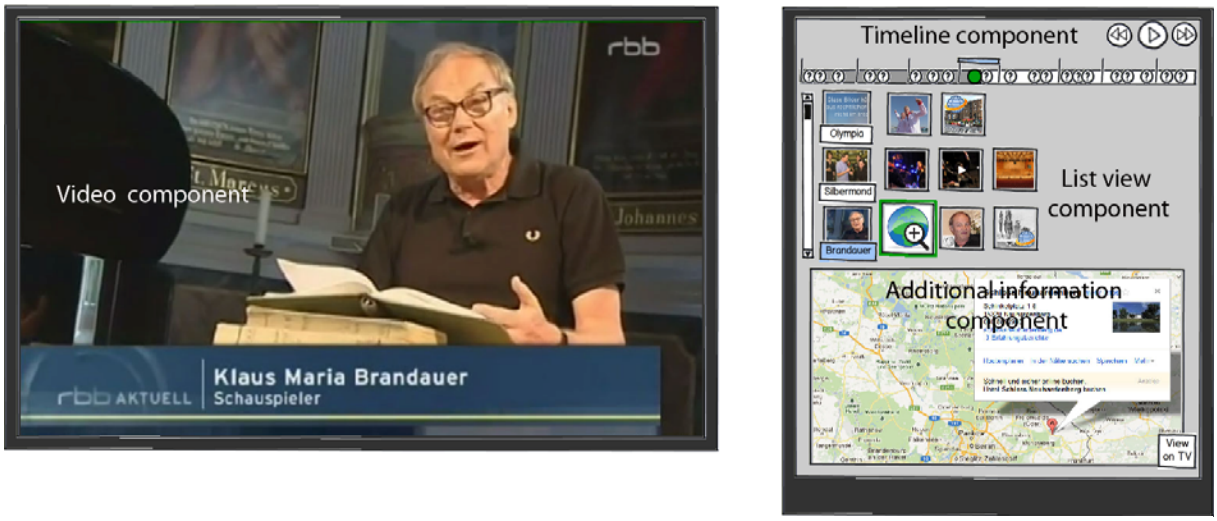

Figure 11: Second screen mock-up– timeline and list view

2.4.2 Circle view mock-ups

The circle view mock-ups contain the same elements as the in the television mock-up. The additional information sources are displayed around the topic. Related videos can be displayed on both the second screen or television screen.

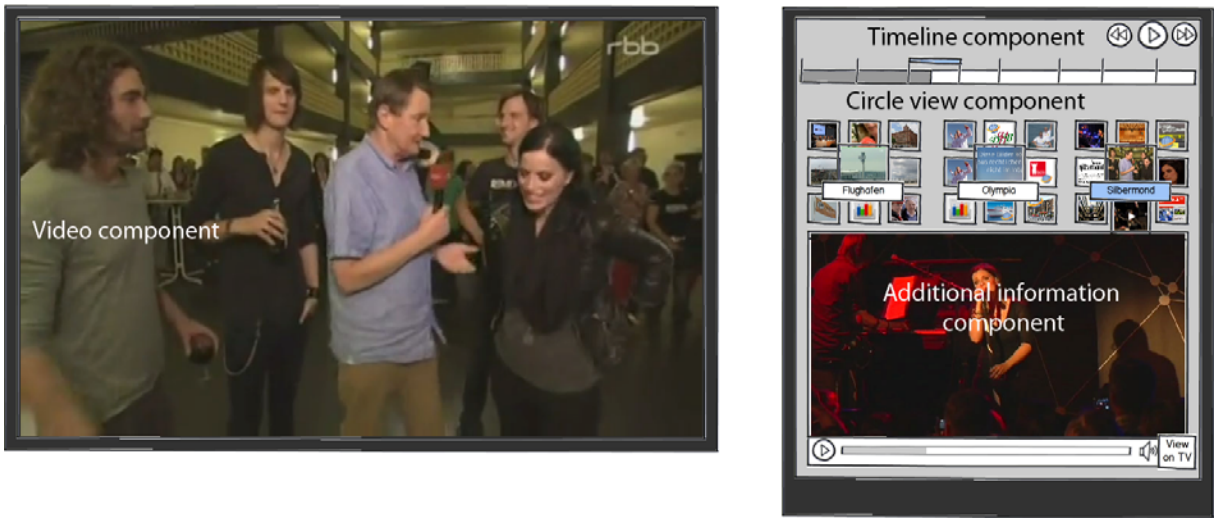

Figure 12: Second screen mock-up – circles

3 User Interface Aspects

While designing for a broad spectrum of devices the user interface should aim to support all of these different devices. Here are the general guidelines taken into account while designing the Interface and Presentation Engine.

3.1 Television UI Aspects

User interfaces for televisions distinguish from other user interfaces by that the viewer is viewing from a much larger distance then with PC monitors or mobile devices. The consequence of this is that the design should consist of larger elements and text should not be too small in order to make it readable, a so called 10 foot UI.

Currently the most employed method of interacting with the user interface is by means of using a remote control. However, these remote controls don't allow easy browsing over the screen, making it important to not have too many elements with which a user can interact; otherwise the user will have to use his remote control too often. Second, all options offered need to be accessible with the keys of the remote control. To allow an easily adoptable system it is wise to give the user a minimum of different keys to use, otherwise the learning curve will be higher and user adaptation of the system may drop.

Another way of interacting with the user interface might be by means of user gestures. These are currently mostly employed on touch-based interfaces such as the Android and IOS interfaces. However, Microsoft has shown that with their Kinect hardware it is possible to detect movements of the user, and attach certain functionality to certain movements (gestures). These gesture might provide a more intuitive method of interacting with the user interface.

Besides this, it is wise to limit the amount of menu options, as these are not common to use on television screens and can distract the user from the video or content that is being viewed. In a scenario where the television screen is the only screen available (no second screen) we should be conscious on the proportions between the data and video. Our assumption is that at least 25% of the screen should still be filled with the video, otherwise the video becomes too small for the user to view. If more space is needed to display data it is preferred that the video will be paused and the data mode goes to fullscreen.

3.2 Second screen UI Aspects

The second screen UI differs most in the way it is controlled, namely by touching the screen. This allows more controlling options then when a user has to use a remote control. However there are some limitations. The size of second screens is smaller compared to television or computer monitors. However since they are only used for the display of additional information the limited size is often sufficient for this purpose. Also the ease a user can scroll through data with a single gesture resolves this issue as long as there is no need to scroll to heavily through the data and interface. Another important aspect with second screens is that

their main purpose is to support the first screen, not to replace it. In other words, it is important to pay attention what to display on the second screen and not overload the user with too much information otherwise the users focus might be more and more on the second screen. Another way to prevent is to allow the user to play additional data or show additional information on the television screen. In such a way the television screen keeps the role of the most important screen where the second screen is there to support and complement it.

4 Presentation Engine structure

The Presentation Engine of the LinkedTV system is the part of the system that combines the content and the meta-data of the presentation with the player into the front-end player for the player.

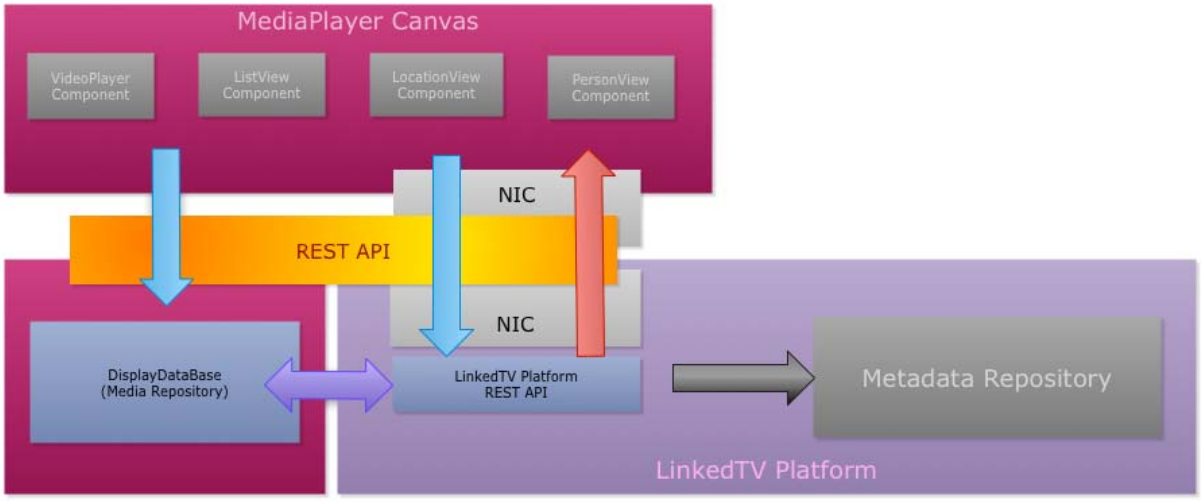

Figure 13: Overview Media player LinkedTV

The above picture shows the architecture of the media player of the LinkedTV platform. The purple part shows the MediaPlayer Canvas. The Presentation Engine consist out of the Media Player Canvas, NIC and the XML containing the information about a specific presentation, which is stored in the Display Database.

In this chapter we will describe briefly the working of the Presentation Engine, the Display Database, the Network Interface Component and the Visual Components in the above picture. A more deliberate technical description of the system van de found in Deliverable 5.2 *Video Player Media Canvas API.*

4.1 Presentation engine

The Presentation Engine needs to be able to handle the different interfaces required for the different scenarios and different devices. In order to do so the engine is designed with a service oriented architecture and service bus structure in mind. SOA gives the possibility to design a flexible interface as the data is uniform available from the different services. The bus structure allows easy expanding of the engine with new components. This all makes that the engine will be scalable and it allows new components to be integrated independently from each other.

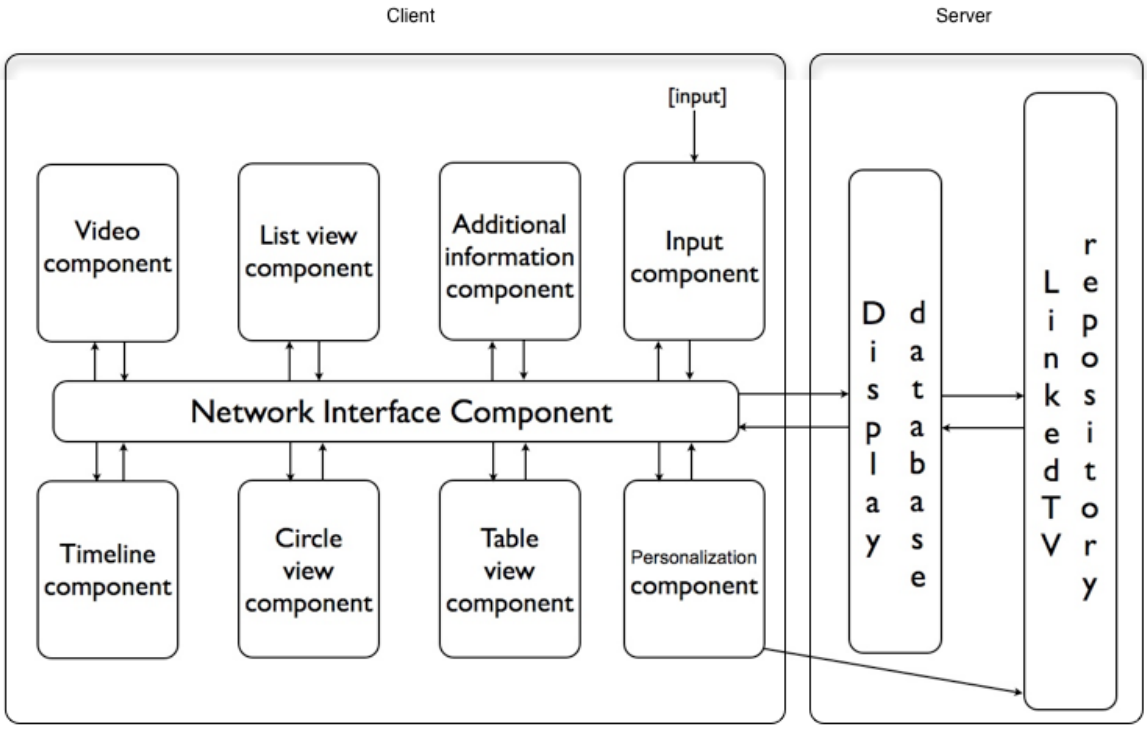

Figure 14: Overview of the Presentation Engine

4.2 Network Interface Component

For the Interface and Presentation Engine the Network Interface Communication component is handling communication between the different components and allows the components to register themselves as part of the interface. By means of this, the components become aware of other components and can adapt or communicate with the other components. The NIC handles all communication between the different components, with each other and with the back-end making it acting like a proxy. Components can send out events, these are send from the component to the NIC, from here the NIC will propagate the event to all components that have registered with the NIC. This way the different components can react by means of for example an update of the interface.

4.3 Presentation XML and display database

The data from the LinkedTV repository back-end services is accessible through a REST interface that returns XML or JSON. This data is not requested directly by the NIC but through the Display Database. This in order to keep compatibility with the current components developed by Noterik. The XML or JSON is parsed in the Display Database to the Noterik file system XML. The data itself is retrieved from the LinkedTV repository by means of the available API's.

Important assumption of the existing Noterik presentation XML is that rather then saying that *a video is played* the concept is based on the idea that a *presentation is played* that can contain more then just video. It could be multiple video's, or video's together with powerpoint slides, al content that is inserted on a specific part in the timeline. Below you find en basic example of such a presentation XML.

```
<fsxml>
  − <presentation id="102">
      − <properties>
        <date_created/>
        <title>30 november 2010</title>
        <description>Raadsvergadering van 30 november 2010</description>
        <liveinput>rtmp://cw2.noterik.com/live/webtv_lelystad_1.mp4</liveinput>
        <polling2>false</polling2>
        <liveinput2>mms://ms21.ntprvgld.prvgld.nl/Gelderland</liveinput2>
        <ipfilter2>212.108.*.*</ipfilter2>
        <livestate>archived</livestate>
        <livecomment>De vergadering begint om 21:30 uur.</livecomment>
        <livetime>Tue Nov 30 2010 09:27:24 PM</livetime>
        <liveendtime>Tue Nov 30 2010 09:27:24 PM</liveendtime>
        </properties>
− <videoplaylist id="1">
    <properties/>
      − <speakers id="408622">
        − <properties>
            <starttime>23196</starttime>
            <duration>105138</duration>
            <time>1291148867903</time>
            <lastname>Horselenberg</lastname>
            <organization>Burgemeester</organization>
            <firstname>Margreet</firstname>
      − <picture>
            http://images1.noterik.com/domain/webtv/user/horselenberg.jpg
      </picture>
      <orgid>100</orgid>
      </properties>
      </speakers>
```

In the above presentation.xml you see the title of the presentation and the that on certain time it will show a speaker in the videoplayer. The main idea behind that format is that every play-out has a timeline and on this timeline your can different types of content, such a video, pictures, text etc but also more interactive elements, such a comment field or a multiple choice question. All this timebased information is stored in the presentation XML of this embed.

4.4 Input Component

Another important part of the Interface and Presentation Engine will be the component that is responsible for the handling of the input. It should be flexible enough to handle the different ways of input, remote control, gestures and mouse or keyboard interaction. To achieve this the input component will have for any kind of input a separate sub component to handle the correct input and map these to actions. So for example the remote button “Play” will result in the same action as pressing the P in the web browsers player. These actions are then propagated to the NIC that propagates the signal to the other components.

4.5 Visual Components

The visual components are the system elements that have a visual appearance on a screen. In chapter 2 we have identified the following component:

- Video component
- List view component
- Additional information component
- Timeline component
- Circle view component
- Table view component

4.6 Personalization component

It is important that WP4, the work-package responsible for personalization, keeps track of the users interaction with the system. This component relies on the input component and will communicate all the actions the user has performed, including metadata about the content this has been selected, on to the server. Feedback is given through AJAX and jQuery, which allows to update the related content to fit the users needs.

4.7 Styling

The visual components should be able to be styled for the different devices since a UI for the television screen has different requirements then a UI for a second screen. In order to do so we want to maintain a clear separation between the data and the styling of data. HTML5 technology allows us to style the interface elements with the help of CSS that can easily be seperated for UI design for the different devices. Based on the device the Interface and Presentation engine should select the correct CSS that then applies the correct styling on the data.

5 Second screen

The focus of the Presentation engine for the second year of the LinkedTV project will be on the integration with the second screen. This makes that the different components of the engine are now divided over two devices. This introduces some challenges that need to be resolved in order to make everything working.

5.1 Synchronization

One of the first hurdles that has to be solved while using the second screen is setting up the initial connection between the first and second screen. There are several ways of accomplishing this. We keep in mind that we try to find a solution where the user should have to do as little as possible to make it easy to setup the synchronization. First, it is possible to have one of the devices discovering the other device by a network protocol for example DLNA. The advantage is that the user has to do very little to establish a connection between two supported devices. The downside of this approach is that both devices should support the DLNA standard.

Currently not every TV and most second screen devices don't support DLNA yet. This makes that the projects goal to support a broad range of different devices cannot be met. Another way of setting up the initial synchronization is by using a QR code. The television shows a QR code that needs to be captured by the second screen. The QR code in mobile devices is used to contain a link that once scanned will open on the mobile device. The advantage is that nearly every second screen device supports this and automatically opens a website. The downside is that in order to use this technology the users second screen device needs a camera.

We prefer the usage of QR codes, first because this will automatically lead the users to the HTML5 application without the need of manually installing or entering a URL. Second, the QR code is currently supported by HbbTV[2] making the integration with this platform much easier.

[2] http://www.rbb-online.de/stadt_land/dossiers/ifa/ifa_2012/aktuell/hbbtv___kann_immer.html

5.2 Remote NIC

To maintain the connection between both devices a central server side component needs to be added that keeps track of actions on both sides. If for example the video is paused on the television the users second screen should also be aware of this and stop the interface from showing timed events that occur later in the video. With the current Interface and Presentation Engine only capable of having the NIC controlling one device the NIC has to be expanded to allow the control over both different devices. For this reason the NIC will be split in two components, one serverside component that keeps track of inter device communication and a local NIC that handles the local communication between the components and communication with the other server side NIC. This has the advantage that the current components don't need to be updated in order to support the second screen. By rewriting part of the NIC code and moving some to the server side the Interface and Presentation Engine will be able to support second screens.

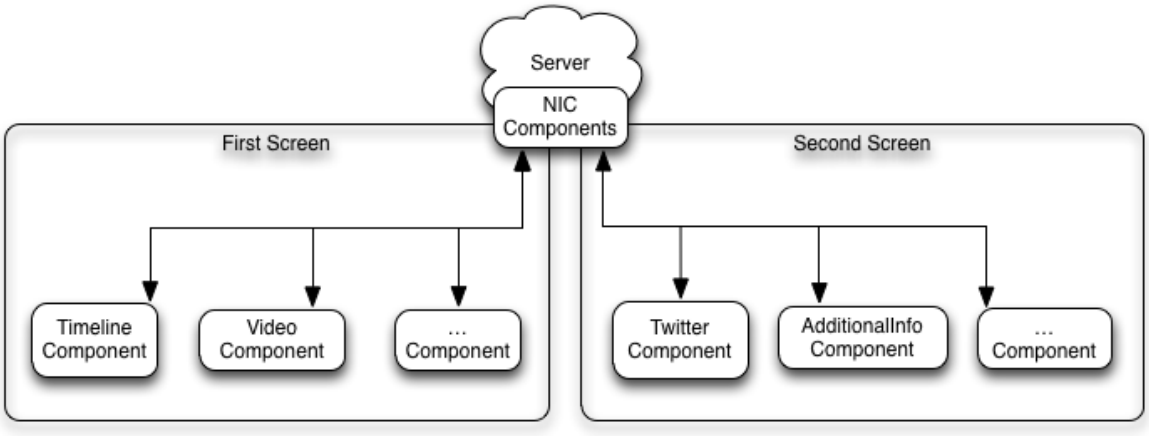

Figure 15: The distributed NIC components

6 Conclusion

Currently, certain parts of the functionality described in this document have been embedded in the LinkedTV HTML5 demo which was demonstrated to the partners. However a lot of this functionality was mocked up.

It is planned to start working on the distributed NIC architecture in order to allow for communication between components in a single or multiple contexts (second screen). The Media Player Canvas API (described in deliverable 5.2) relies on the distributed NIC architecture to be operational in order to be communicate with the components running in the context of the application. Therefor the functionality embedded within the NIC will first have to be developed.

For the coming year, we plan on further integrating the distributed NIC and the Media Player Canvas with each other.

7 References

1. Google, Designing For TV, https://developers.google.com/tv/web/docs/design_for_tv
2. jQuery, http://jquery.com/
3. Digital Living Network Alliance, http://www.dlna.org/