Digital Signage System

ABDUR RAUF KHAN

KTH Information and Communication Technology

Master of Science Thesis
Stockholm, Sweden 2009

TRITA-ICT-EX-2009:207

Digital Signage System

Abdur Rauf Khan
rkjadoon@kth.se

Masters Thesis

2009-11-19

Supervisor & Examiner

Professor Gerald Q. Maguire Jr.

School of Information and Communication Technology
Royal Institute of Technology (KTH)

Stockholm, Sweden

Preface

The work described in this report was carried out at the Wireless@KTH the Center for Wireless Systems at the Royal Institute of Technology (KTH), in Stockholm, Sweden. The major goal of the project was to design, develop, and evaluate a centralized system, based on a PC to display web contents (or other dynamic information) on digital displays located at one or more locations. Our target display was the large screen TV/display located at the entrance to the Wireless@KTH center in the Electrum building in Kista.

Abstract

Digital signage is an emerging new communication technology. It is expected to play an important role in today's dynamic world as digital signage displays timely information, while reducing the environmental costs associated with traditional printed signage. The main focus of this thesis is the design, implementation, and evaluation of a digital signage system based on a PC to display web contents (or other dynamic information) on digital displays located at one or more locations.

The system will display information based upon a "playlist" that can be dynamically updated. It avoids the single point of failure of a television (be it analog or IPTV) based system, since each display has an attached processor and local storage containing both the information to be displayed and the local playlist of what is to be displayed. Additionally, the design allows content to be customized to specific local viewers, i.e., the information displayed can be adapted to the user or users currently in front of the display. The granularity of the schedule (i.e., playlist) is much shorter than in existing digital signage systems – leading to a more visually dynamic experience for viewers. On the basis of our evaluation, we strongly believe that this approach to digital signage will displace existing signage systems.

Sammanfattning

Digital signage en framväxande ny kommunikationsteknik spelar en viktig roll i dagens dynamiska värld. Digital signage visar värdefull och aktuell information för utbildning, näringslivsorganisationer och är viktigt för samhället, eftersom det ger både ett sätt att ge aktuell information och minska de miljömässiga kostnaderna i samband med traditionella tryckta skyltar. Tyngdpunkten i denna avhandling är att designa, implementera och utvärdera en digital signage system baserat på en PC för att visa webb-innehåll (eller annan dynamisk information) om digitala displayer som finns på en eller flera platser. Systemet kommer att generera värdefull dynamisk information i rätt tid till en grupp av olika tittare. Svarstiden för att visa innehållet på den digitala displayen är mycket snabb jämfört med andra skyltsystem. På grundval av vår utvärdering, tror vi starkt att den kan ersätta det finns skyltsystem på marknaden.

Acknowledgements

I would like to sincerely and heartily thank my examiner, Professor Gerald Q. Maguire Jr., for his kind help, considerable encouragement, great suggestions, and the opportunity he gave me to work in his lab to complete my masters thesis project. His positive attitude and excellent insights in my work and life rewarded me with a valuable experience at Wireless@KTH.

I would like to thank my friends and family for their sincere encouragement and help in both my life and thesis.

Lastly, I offer my regards and blessings to all of those who supported me in any respect to complete the project.

Table of Contents

List of Figures

List of Tables

List of Abbreviations and Acronyms

AVI	Audio Video Interleave
CDMS	Cisco Digital Media System
CF	Compact Flash Secure Digital
DMP	Digital Video Broadcasting Terrestrial
GUI	Graphical User Interface
HD	High Definition
HDD	Hard Disk Drive
IPTV	Internet Protocol Television
ISO	Internet Standard Organization
LDAP	Light Weight Directory Access Protocol
MPEG	Moving Expert Picture Group
MRBS	Meeting Room Booking System
SD	Secure Digital
SOAP	Simple Object Access Protocol
WAE	Wide Area Application
WMV	Windows Media Video
WSDL	Web service Description Language

Chapter 1 - Introduction

1.1 Introduction

Today, digital media is the most compelling platform to effectively reach employees, students, customers, and partners. This digital media is used to convey important information and messages such as news, training material, and information about upcoming or current events. Digital media is effective because it brings familiarity and closeness to modern communications [1]. In today's dynamic world digital media has an important role; especially for organizations who wish to spread their business throughout the world. For educational organizations digital media plays an important role in informing students about the events, such as: seminars, lectures, meetings, registration deadlines, schedule changes, exams, and sports activities. Digital media represents an emerging new communication technology; in particular digital signage is rapidly gaining popularity today.

Digital signage is emerging as a new communication technology. A digital sign is defined as an electronic display that shows information, advertising, or other messages [2]. "Digital signage is a network of customizable displays that can be controlled electronically using a computer, allowing content to be changed remotely for the most targeted messaging possible"[3]. Digital signage can be implemented using liquid crystal displays, light emitting diodes, digital projection, plasma displays, etc. Such digital signage can be used in airports, research organizations, shopping malls, railway stations, and restaurants to dynamically deliver information, graphics, animations, videos, text, and other web contents on a (high quality) display to targeted viewers at a specific time.

The main challenge today for many organizations is the successful deployment and integration of a digital signage network system. In large organizations, such as business, educational, and research organizations, lots of events are taking place each day. Thus there is a constant need to inform people about what activities are taking place, where they are taking place, and how to get from where the sign is to this place. Informing the potential audience in these organizations through conventional signage has many disadvantages and drawbacks. Some of these drawbacks and disadvantages are the financial costs of printing, distributing, and removing paper posters; the environmental costs of these poster over their lifecycle; the fixed contents of such printed posters (i.e., any change is expensive as it requires a physical change to the poster - once it is printed); printed posters can only convey static text and pictures.

As an example of a digital signage networking system, consider the Cisco Digital Media System [4]. This digital media system consists of the digital media player and IP attached endpoints that can display high definition (live and on demand) video, web pages, graphics, animations, text, and other dynamic content on digital displays. This system supports both digital signage and enterprise TV.

In this research project I designed and developed a prototype system, based on using a PC directly attached to a digital display in order to display web contents (or other dynamic information). Our target display was the large screen TV/display located at the Wireless@KTH center in the Electrum building in Kista.

1.2 Objective and Focus of the Thesis

The main focus and objective of this thesis was to design, develop, and evaluate a hardware/software solution based on a PC, to control and display different web contents (or other dynamic information) on digital displays. Instances of this system could be located at different locations.

The system will display information based upon a "playlist" that can be dynamically updated. A secondary goal of this research project was to remove the single point of failure of a television (be it analog or IPTV) based system, since each display has an attached processor and local storage containing both the information to be displayed and the local playlist of what is to be displayed. In existing TV based systems when the main server crashes the whole system stops working.

This thesis project will also explore how to leverage existing sources of information, for example those that are available via the intranet or internet. Additionally, the design should allow content to be customized to specific local viewers, i.e., to enable the information to be personalized to an individual or group of people gathered in viewing distance of the display. Thus an access control mechanism is needed to allow different users and groups, to dynamically modify the information that is being displayed.

More generally this project will explicitly consider how to generate a cycle of information to be displayed, where the information can be adapted based on time, location, and viewers. The granularity of the cycle's schedule (i.e., playlist) is much shorter than that of existing digital signage systems – leading to a more visually dynamic experience for viewers.

1.3 Organization of the Thesis

Chapter 1 gave a basic introduction to digital signage and the main objective of this thesis project. Chapter 2 - introduces some background material related to digital signage. Chapter 3 - describes some of the existing digital media players, while Chapter 4 - describes the existing network architectures used for digital signage. Chapter 5 - presents a brief market analysis. In Chapter 6 - we present some related work regarding digital signage. Chapter 7 - presents the overall design and implementation of our project, while Chapter 8 - describes the details of our prototype. An evaluation of this prototype is given in Chapter 9 - .The thesis concludes with a summary of our conclusion and suggests for future work in 9.4.

Chapter 2 - Background

In this section we summarize and elaborate upon the basic concepts and technologies which we think are important and valuable to our research.

2.1 Conventional Signage

Conventional signage is a widespread and traditional advertising method that is used to deliver different messages and information in specific locations. There are different groups of advertisers/companies working to deliver conventional signage. These advertisers use large numbers of signs both indoors and outdoors to broadcast their messages. Traditional signage is used near roads, markets, large shopping centers, bus stops, airports, railway stations, casino, and hotels. These signs are typically made from plastic, steel, aluminum, vinyl, glass, and paper.

Conventional signage has many disadvantages in terms of its construction, cost, distribution, and installation. The main disadvantages of traditional signage are that it is time consuming and expensive to create, distribute, and install. To make and install a sign requires passing through a series of different teams and processes. More explicitly, once a graphic designer or advertising expert makes a sign, he or she will deliver it to the distribution team who are responsible to distribute this sign to its proper location and deliver it to the installation team. The installation team installs the sign in its location. The cost of signs includes material costs, human resources, and environmental costs (creating the sign, distributing it, installing it, and removing it) leading to high cost and generally requiring a substantial budget to deploy a large number of signs or to have signs in high traffic locations.

The second disadvantage of conventional signage is that it can only display static messages and information. Thus a single sign can display only one message or advertisement at a time (although some signs are motorized to display a small number of signs in a cycle). These static signs cannot advertise or display more dynamic information, such as web contents, videos, and advanced graphical animations. Displaying different types of information for specific people will be very useful. For example, an advertisement in a shopping mall might be displayed during certain hours and targeted to older people, but late in the evening there might be many teenagers and young people in the shopping mall, thus the display should be adapted for these specific viewers. However, this is very difficult and expensive with a conventional signage system.

2.2 Digital Signage

Digital signage is a wide spread solution allowing reliable centralized management and publishing of digital media to networked, digital displays. This technology is now gaining popularity and becoming more ubiquitous. Digital signage has many advantages over traditional signage. It can display more dynamic messages, videos, animation, web contents, and even real-time video broadcasts. This dynamic signage can attract and retain more viewers. Digital signage consists of a display device and a display controller. This display controller can be a PC or digital media player. A simple digital sign display is shown in Figure 2-1.

Many organizations are now changing from conventional signage to digital signage. This digital signage gives both benefits and advantages to the organization, enabling them

to extend their business and to attract more customers. For educational organizations such as universities and colleges digital signage can provide useful information to students, faculty, staff, visitors, etc. about the events and provide other important information.

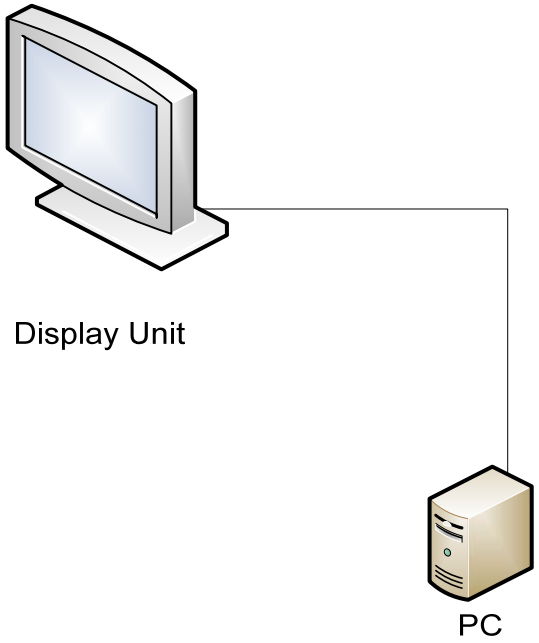

Figure 2-1: The simplest form of Digital Signage

Digital signage reduces the cost of construction and installation; as there is no need to fabricate a new sign for each different instance of content that is to be displayed, and it also reduces the cost of distribution and installation. There is in many cases no need to manually distribute and install the signs. Nor is there a need to remove out of date signs.

2.2.1 Types of Digital Signage

There are three main types of digital signage:

- Stand-alone Digital Signage
- Web-based Digital Signage
- IPTV-based Digital Signage

2.2.2 Stand-alone Digital Signage

Stand-alone digital signage deployment is very simple as was shown in Figure 2-1. It consists of a display and a computer. The computer is not connected to a network. The contents of the digital display(s) will be controlled by this computer. New display contents can be installed using a USB memory stick or other portable storage device. A simple stand-alone digital sign might only display a single fixed display of content or it might change different parts of the display based upon a static schedule.

Advantages

- Deployment is easy and simple.
- Cost effective. No cost for network connectivity.
- Can be managed and controlled by an employer or user.

Disadvantages

- Provides only display of semi-static contents (i.e., when the contents are loaded - the contents are fixed until someone physically comes to the computer and loads new contents). However, as noted above the contents could be display in a time dependent manner.
- The lack of network connectivity limits the information that can be provided to be display to semi-static contents that are known when the contents are written to the storage device for later loading into the computer.
- Can not be used to deliver information that changes (in an unpredictable way) faster than the time between loading new content into the computer.

2.3 Web-based digital signage

In this type of digital signage, the contents of the signage/display can be directly controlled by a local web browser (see Figure 2-2). All the contents on the display(s) can be easily controlled by users/administrators from anywhere on the network that is able to reach this device (either directly or through a proxy).

Advantages

- No server architecture is required.
- All of the content management can be done via a web browser from a network attached computer.
- It is a scalable solution, since the rendering of the content is done locally - allowing each sign to display information specific to this display.
- Can support different accounts with different access - enabling different types of users and administrators to have different rights with respect to controlling the device and the contents that are displayed.
- Simple and easy deployment with remote access from any browser.
- Cost effective. No need for expensive software or hardware (aside from the cost of the digital sign, associated computer, network connectivity, and electricity).

Disadvantages

- Preferred by small organizations.
- May need high bandwidth, between the computer and the display, but only if full motion video and advanced graphics are desired.

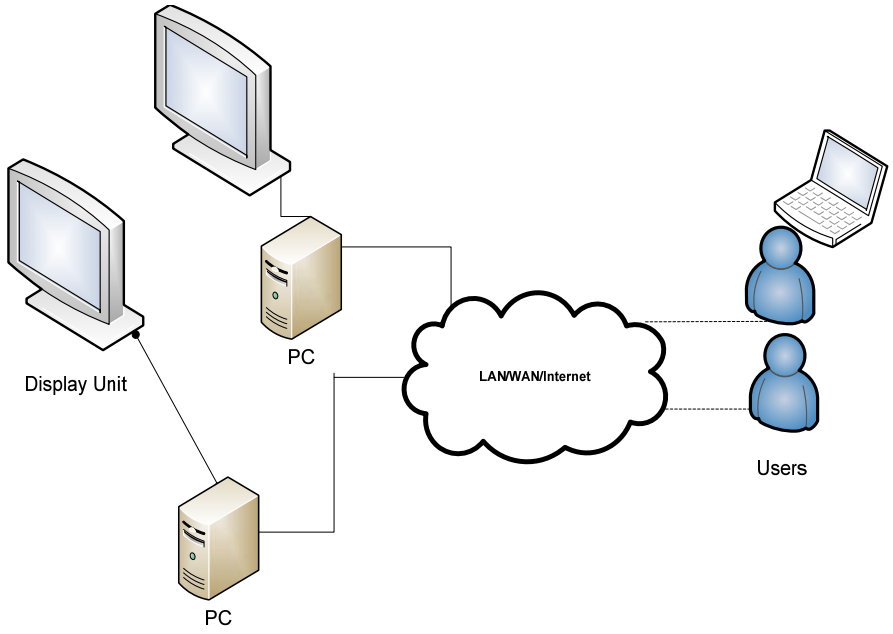

Figure 2-2: Web-Based Digital Signage

2.3.1 IPTV–Based Approach

In the IPTV approach all of the contents are distributed by a streaming IPTV media server (see Figure 2-3). In this approach contents such as video, graphics, animations, images files, and web contents are directly distributed from a central media server to media players attached to networked displays. A central network manager handles the task of organizing, managing, and distributing content to a media player connected to each display. This approach is very suitable for a large number of displays that will display only a limited number of different streams of content. This approach could be viewed as a closed IPTV service, where different displays can select different IPTV "channels".

Advantages

- SRTP or other techniques can be used to secure the content and provide data integrity.
- Due to the commercial use of IPTV, IPTV services can be highly stable and have high available.
- Provide advanced features such as unlimited channel capacity (unlike analog TV based distribution which offers only a finite number of channels of capacity).

Disadvantages

- Large and complex network infrastructure i.e. requires installations of servers, deployment of media players, and management of media services.
- This type of solution is based on PCs, servers, and media players. Some of these hardware components may be relatively inexpensive (for example, media players), while other components can be more expensive (such as media services to support a large number of different media streams).
- Training and integration with databases may be needed.
- Needs high bandwidth between the media service and the media players.
- If a central server fails, then all the system is unavailable.

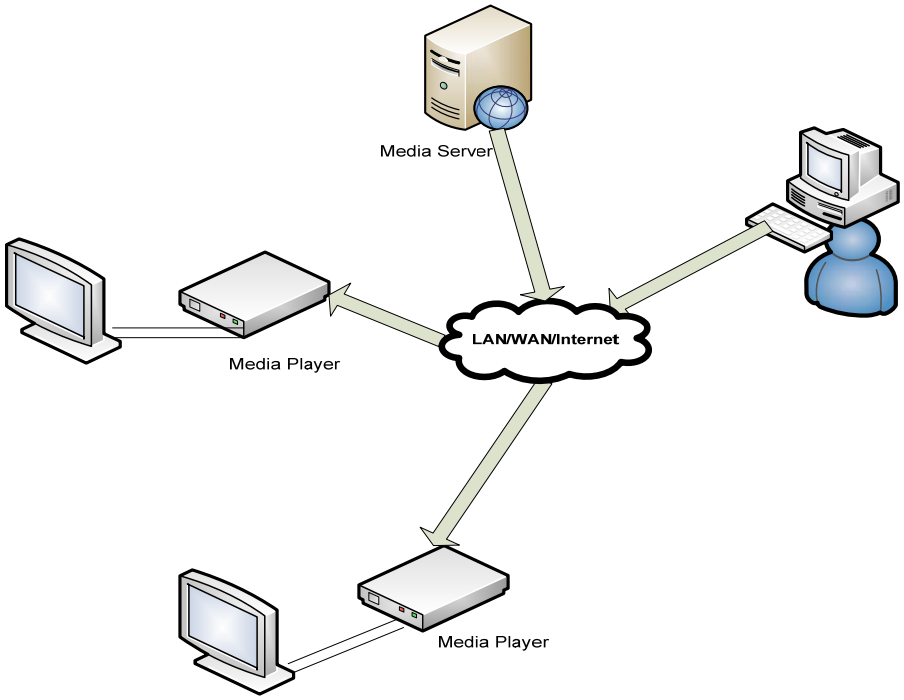

Figure 2-3: IP-TV based approach to digital signage

2.3.2 Comparison between types of digital signage

A comparison between these three types of digital signage in terms of cost, complexity, management expertise, scalability network, bandwidth requirements, and browser access are shown in Table 2-1.

Table 2-1: Comparison of different types of Digital Signage

Feature	Stand alone	Web-Based	IPTV-Based
Total System Cost	Low	Low	High
Complexity	Low	Low	High
Management Expertise	Low	Low	High
Scalability	Low	High	High
Bandwidth Requirements	None	High	High
Browser Access	No	Yes	No

These three types of digital signage represent different techniques and approaches to digital signage. After comparing all three approaches, I observed that the deployment of each approach depends on the organizations requirements. Each approach has different benefits and limitations. The approach to digital signage which I have selected to use in this research project is the web based approach, because it is a simple and very scalable approach for a small organization (in this case a research center & department). Based upon this approach we will develop a hardware/software solution based on a PC with a web

browser. The software will be used to control the display and to display different web contents (or other dynamic information) on digital displays.

A centralized system management application will control all the PCs connected to digital displays. This application will allow centralized configuration and management, while allowing the content of each display to be locally generated, thus avoiding the main problem in an IPTV based approach (i.e., when a central server fails all the system will be unavailable). In the web based approach, if the system management hardware/software fails, the network attached PCs and their displays will continue to operate, display contents according to their own schedules, hence continuing to display content on the displays. The main impairment of the central management system's failure will be that coordinated tasks, such as configuration and central distribution of new content (or scheduling of new content for display), will not be possible.

2.4 Digital Signage Applications

Financial organizations: Many financial organizations want to expand their business training and communication out to their many branch offices. They may also want to keep their customers and employees up to date by broadcasting a variety of information through digital signage at their branch locations. In addition to important information, they can also promote their products to increase their sales and improve their customers' experience.

Educational organizations: Digital signage has a very important role in educational organizations to inform their employees and students about upcoming events and to distribute information. It improves campus communications and facilities emergency notifications. It can inform students about upcoming seminars, registration deadlines, exams, and sports activities.

Airports & railway stations: Digital Signage provides information about flights, trains or bus schedules, changes in schedules, arrival and departure information, delays, and other information.

Hotels & restaurants: Communication with hotel guests. Providing up to date and useful information such as special offers, restaurant menus, entertainment announcements, and conference and seminar announcements.

2.5 Content types and management

Digital signage content can include text, graphics, animations, audio, video, and interactive content [2]. Digital signage relies on appropriate content if it is to work effective [2][5]. The type of media or content is very important as it should match the type of digital signage that is to be utilized. Digital signage may support high quality videos, images, and text to deliver compelling messages to viewers. The content which you are going to display must be relevant for your audience. If the contents are not relevant to viewers, then the digital signage will not yield benefits for the organization.

The next section will discuss first the content or media formats that are supported by digital signage, and then it will describe some digital signage management methods. To provide a solid basis for understanding what the requirements of both devices and networking will be, we begin with a description of some video and image formats which may be supported by digital signage.

2.5.1 Content Types

MPEG: MPEG is a Moving Picture Experts Group subcommittee of the International Standard organizations (ISO) for audio, video compression and transmission [6]. MPEG was developed to encode and compression analog sources into digital formats that can be easily stored and distributed.

MPEG1 was the first widely used compression method for audio and video. MPEG1 includes the popular layer 3 audio compression format (referred to as MP3).

MPEG2 is used to broadcast audio and video for television programming. It is the most popular format for distribution of full frame video over IP. It supports compression of High-Definition (HD) video contents.

MPEG3 and MPEG4 were developed to support and achieve greater compression.

Adobe Flash: Adobe Flash is a multimedia platform originally developed by Macromedia. It is currently being developed and distributed by Adobe systems [7]. Flash is used to add animations and graphics to web pages to integrate video into web pages.

JPEG: JPEG is the most common method used for compression of photographic images for storage or transmission over the internet.

2.5.2 Content Management

The management of content is very important in order to display useful and important information to the targeted viewers at a specific time. This is especially important when more than one message is to be displayed, for example, in order to target different viewers. Several content management methods are described below.

2.5.2.1 Dividing screens into regions or zones

To display multiple types of information simultaneously on a digital display, the screen can be divided into multiple regions or zones. Each zone or region can support multiple elements of content and these different elements can even use different formats. Each and every zone or region is an independent component that might be updated or changed independently.

Figure 2-4 shows multiple regions in which different information is shown. In this case: video streaming of news, graphical animation giving weather report, and some textual information (stock prices and volumes) about different companies and the time of day.

Figure 2-4: Display showing multiple zones [8] (Appears here with the permission of Taxes Digital*)

2.5.2.2 Scheduling Content

Scheduling contents of the digital signage enables specific content to be displayed at a specific time of the day or on different days. One of the most important benefits of digital signage is that it can support managed and scheduled contents. For example, an educational organization may want to display some important information to employees at a specific time, while at a different time they want to display information about student activities.

2.6 Conventional Signage versus Digital Signage

A comparison between conventional signage and digital signage is shown in Table 2-2. Another feature that motivates our use of digital signage is the potentially lower environmental impact of digital signage.

Table 2-2: Comparison between Conventional Signage and Digital Signage

Conventional Signage	Digital Signage
Conventional signage can display only static content.	Digital signage can display dynamic information and advertisements.
Cannot support, video, audio, flash animation, and web contents.	Can support, video, audio, graphics, animations, and web contents.
Conventional signage requires a team of humans to distribute and install the signs once they have been produced.	Digital content can be controlled centrally by any authorized employee of an organization.
Cannot support content scheduling, zones, or regions.	Can support content scheduling. The screen can be divided into zones/regions to simultaneously display multiple elements of information.

* Taxes Digital System, www.txdigital.com

Chapter 3 - Digital Media Players

A digital media player is a highly reliable IP based end-point that can control digital signage, including rendering high definition live broadcasts, on demand video, web pages, graphics, animations, text, and other dynamic content on to digital displays. A digital media player has a very important role in managing and controlling digital signage. A number of different organizations make their own media players. These digital media players have different functions, in order to best meet customers (perceived) requirements. Cisco, CAYIN technology, Qumu, and Media Star Evolution are major players in this market.

The following sections will discuss a number of different types of digital media players; along with some of their advantages and disadvantages. The following chapter will describe how these vendors plan for their customers to deploy their network infrastructure to support digital signage.

3.1 Cisco Digital Media Player

Cisco's digital media player is an integrated component of their Cisco digital media system [9]. The Cisco digital media player (shown in Figure 2.5) is designed to be a reliable, dynamic, digital media player. This player can decode and display digital media on digital displays. It supports many types of content including: high-definition live broadcast, on demand video, flash animations, text tickers, and other web contents. It allows the programmer to control the full screen as well as define and separately control different regions and zones. The Cisco digital media player has an RS-232 interface for controlling some types of digital displays (for example, a number of flat screen televisions allow RS-232 input to control the display's configuration and operations). This media player also has a build in graphical user interface (GUI) for device and content playback management. This digital player can be used to implement an IPTV approach to digital signage.

Figure 3-1: Cisco Digital Media Player [9]. (Appears with the permission of Cisco[†])

Models

Currently Cisco has two models of their digital media player [10]:

- Cisco Digital Media Player 4400G
- Cisco Digital Media Player 4305G

[†] Cisco, www.cisco.com

3.1.1 Cisco digital media player 4400G

The Cisco digital media player 4400G has the following features [11]:

- Reliable, flexible, real time publishing.
- Can support live on demand video, flash animations, Adobe flash player 9, and advanced graphics.
- IP based delivery of the live broadcast and live on demand video.
- Easy and rapid deployment.
- Can support MPEG1, 2, and 4.
- 4GB of local storage for reliability and performance.
- Low power consumption: 15W.
- Small form factor: 10″×8″ ×2″ inches (~25 x 20 x 5 cm), weight: 2 kg.

3.1.2 Cisco Digital Media Player 4305G

This player has the following features [12]:

- Can support IP base delivery of the contents.
- Local storage is 2GB for reliability and performance.
- Can support MPEG1, 2, and 4. Can also support web contents, and Adobe flash player 7.
- Lower power consumption: 5W.
- Small form factor: 7.5″×5″ ×1.5″ inches (~19 x 13 x 4 cm), weight: 0.45kg.

3.2 Cisco Digital Media Manager

Cisco's digital media manager [13] offers a web based interface through which the users or content owners can control, manage, and display contents such as videos, texts, graphics, animations, enterprise TV, and desktop video. This web interface is shown in Figure 3-2. This centralized web based application is an integrated part of the Cisco digital media management solution. The main features of this manager are:

- Content managers or users can manage content asset and create playlists for digital signage and desktop video.
- Can schedule future content deployments and playback schedules for digital signage, desktop video, and enterprise TV.
- Allows the user to synchronize slides for both live and video on demand desktop video events.
- Manage user accounts based on access control mechanism and to configure user specific content restrictions for desktop video users through LDAP/Active directory.
- To control, configure, and manage a digital signage network remotely.
- Customize signage screen layouts and zones.

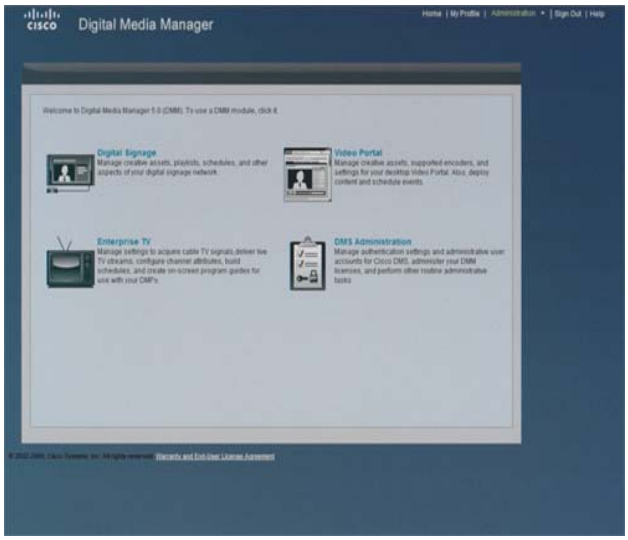

Figure 3-2: Cisco Digital Media Manager Web interface[13]. (Appears with the permission of Cisco)

3.3 Stinova Digital Media Players

Stinova Ltd. [14] is a subsidiary of STINO media AG (Holding). The company is a worldwide sales and marketing organization for digital signage, IPTV, and other video, & audio solutions. This section will discuss a number of different types of Stinova digital media players.

Stinova has introduced many digital media player for different purposes such as DMP 6500, 6600HD, DMP 6700, 6800, 7900, 9432, 9440, and 9546[15].

DMP 6500 is a cost effective media player with 2GB of internal memory, which is capable of displaying many types of contents for digital signage displays. It features a low power consumption of 22 Watts.

DMP 600HD supports full HD video contents for digital displays. This player is designed to be cost effective and have low power consumption (9 Watts).

DMP 6700 (Celeron based), **DMP 6800** (Core 2 Duo base) are cost effective media players having optional 80GB HDD (Hard Disk Drive) for a variety of digital applications.

DMP 7900 is a highly flexible media player based on a combined PC and setup box architecture in one device. This device supports full HD video contents and many other contents formats at the same time. This device also supports DVB-T to enable over the air reception of programming.

DMP 9432, DMP 9440, and DMP 9546 are a series of new generation, high end modern industrial grade player targeted at 32” and 44” TFT display. These devices are based on an Intel platform with 1GB RAM, modern graphics, supporting 90° rotated portrait mode display and a slot for CF (Compact Flash) for flash memory. It is possible to manage the display panel with digital media server software; including the ability to schedule information play out via a web browser. Extension to DVB-T may also be possible.

3.4 Cabletime's Media Star Digital Media Player

Cabletime's Media Star Evolution [16] introduced a model 780 HD/SD media player and a HD/SD IP TV receiver (shown in Figure 3-3). Either device can be used as a digital signage media player to deliver high definition TV, multimedia contents, and web contents to any HD digital display screen. The features of these devices are:

- Both of these media players have a 4GB internal memory and support the latest video compression standard (MPEG-4 AVC/H.264). They have the ability to deliver both live HD, SD MPEG-4, AVC/H.264, and MPEG-2 video streams and locally stored HD, SD MPEG-4, AVC/H.264, and MPEG-2 video streams to/from an on board internal memory.
- Ability to show slide presentations based on HTML, JPEG, PNGs, and can show web pages.
- It can reliably operate its own content or can be networked to provide automated content download and synchronized video contents with other 780 HD/SD units.
- It has browser/RS232/USB and IR remote control interface built on a reliable dual core computing engine running under Linux.
- It can be operated standalone or under the control of Media Star administration software.

Figure 3-3: Media Star-780/4GB HD/SD Digital Signage & IPTV Receiver [16].
(Appears with the permission of Cabletime[‡])

3.5 CAYIN Digital Media Players

CAYIN technology [17] introduced dynamic digital signage solutions including digital media players, servers, and management software. CAYIN introduced a number of different media players. Here we will mention some of these media players along with their features, support, and applications.

SMP-PRO3 and **SMP-PRO3N** can support 6 zones on the display; and can display videos, tickers[§], and clocks. It can support one background design, one multimedia (stored video, streaming video, or audio video input) area; two image slide show areas, one ticker area, one clock, and date area. They provide an editor tool though which a user can easily define the size and location of each zone. The devices support a number of video CODECs and image file formats including: MPEG 1/2/4, AVI, WMV 7/8/9, WMA, raw DV, JPEG, GIF, and MP3.

The **SMP-PROPLUS** is a high performance zone type signage player supporting a maximum of seven zones and portrait display mode. It can support one background design, two multimedia (stored video, streaming video, or audio video input) areas; two image slide show areas, one ticker area, one clock, and a date area. It has a web based user

[‡] Cabletime, http://www.cabletime.com/
[§] A ticker in this setting is left to right scrolling text, similar to the traditional paper stock tickers.

interface through which users can easily control the contents of digital display. It supports the same CODECs as supported by the SMP-PRO3 and SMP-PRO3N.

SMP-WE3/SMP-WE3N signage players support display of full screen HTML, flash, and JPG; as well as stored or real time video playback. The devices support web based presentations and most popular web programming languages. Both types of devices provide a web based interface through which the player can be easily controlled and managed remotely. It has the ability to play real-time video or display an image when connecting to a TV tuner, DVD player, or camera. The most important feature is that it is able to open a HTML file from the player's hard drive as well as from a remote URL. It supports the same CODECs as supported by their SMP-PROPLUS media player.

The **SMP-WEBPLUS/SMP-WEBPLUS-T** are both designed to support highly flexible multimedia presentations and when deployed under an open structure can easily be integrated with applications. The devices have the ability to support most popular web languages both portrait & landscape presentations. The devices support HD video, HTML, web page images, and text. A distinguishing features of these devices is the ability to play **two** videos at the same time. A web based user interface allows easy remote control of each media player.

The **SMP-WEBDUO** signage player is the most advanced type of CAYIN digital media player. Each player can support two screens at the same time. The player has a dual core processor and can support both single and dual displays in portrait or landscape screen with high quality 1080P full HD video.

3.6 Summary: Media Players

Media players play an important role in the successful deployment of digital signage. Each and every media player has different features and support. Each company makes different types & models of media player - each with different choices of features such as the amount of internal memory, available CODECS, system power consumption, support for HD, and zoning support. Summarizes a number of different media players.

Table 3-1: Summary of Media Players

Company	Cisco	Stinova Ltd	Cabletime	CAYIN Technology
Models	4400G, 4305G	DMP6500, DMP600HD, DMP6700, DMP6800, DMP7900, DMP9432, DMP9440 and DMP9546	Media Star Evoloution 780	SMP-PRO3, SMP-PRO3W, SMP-PROPLUS, SMP-WE3, SMP-WE3N, SMP-WEBPLUS, SMP- WEBPLUST and SMP-WEBDUO
Internal Memory	4GB, 2GB	2GB, 80 GB depending on model	4GB	80GB, 160GB
CODECS	MPEG1/2/4, Adobe flash 7, 9	Full HD, Adobe flash 9, Advanced graphics	HTML, JPEG, PNGs, HD, SD, MPEG4	MPEG 1/2/4, AVI,WMV 7/8/9, WMA, RAW DV, JPEG, GIF and MP3
Power Consumption	15W, 5W	22 watt, 9Watt depending on model	7W	7W
Full HD Support	Yes	Yes (All support Full HD)	Yes	Yes, full support for all models
Zoning Support	Yes	Yes (All support Zoning)	Yes	6 to 7 zones for all models

Chapter 4 - Network Architecture

The most important challenge for both IPTV and web based architectures is how to integrate digital signage deployment with the organization's network infrastructure. Alternatively one could ask the question: How can we use the organization's network infrastructure to deploy digital signage easily? This section will discuss some deployment techniques. It will also present how to setup both the Cisco and CAYIN digital displays.

4.1 Cisco Digital Media Network Architecture

Cisco's digital media system architecture is based on the Cisco Wide Area Application Engine (WAE)[1]. This application engine is able to automatically and reliably distribute and stream digital media content. Cisco's digital media system provides a secure high performance implementation with the help of their WAE and their Cisco Content Networking System Software. Cisco's digital media system network architecture is shown in Figure 4-1. It provides the following features

- Supports both live unicast and multicast streaming services.
- Provides on demand access to video and audio (for viewing at LAN speeds) or files cached locally.
- Reduces video bandwidth to minimize its effect on network traffic.
- Securely and efficiently distribute contents over the network.

Cisco's digital media system supports applications such as digital signage, enterprise TV, and Cisco Desktop Video. The media can be web contents, video, audio, and graphical animations. These different applications use different combinations of hardware & software, specifically media encoders, digital media players, and a digital media manager.

Media Encoders are used to capture and digitize media from different sources and convert them into digital formats for live and on demand delivery across an IP network. Digital media players (as described in the previous chapter) support different types of content including high-definition live broadcast, on demand video, flash animations, text tickers, and other web contents. The digital media player allows us to control the full screen, as well as different regions and zones in devices that support regions/zones. The Cisco digital media manager is a web based interface through which the users or content owners can control, manage, and display contents.

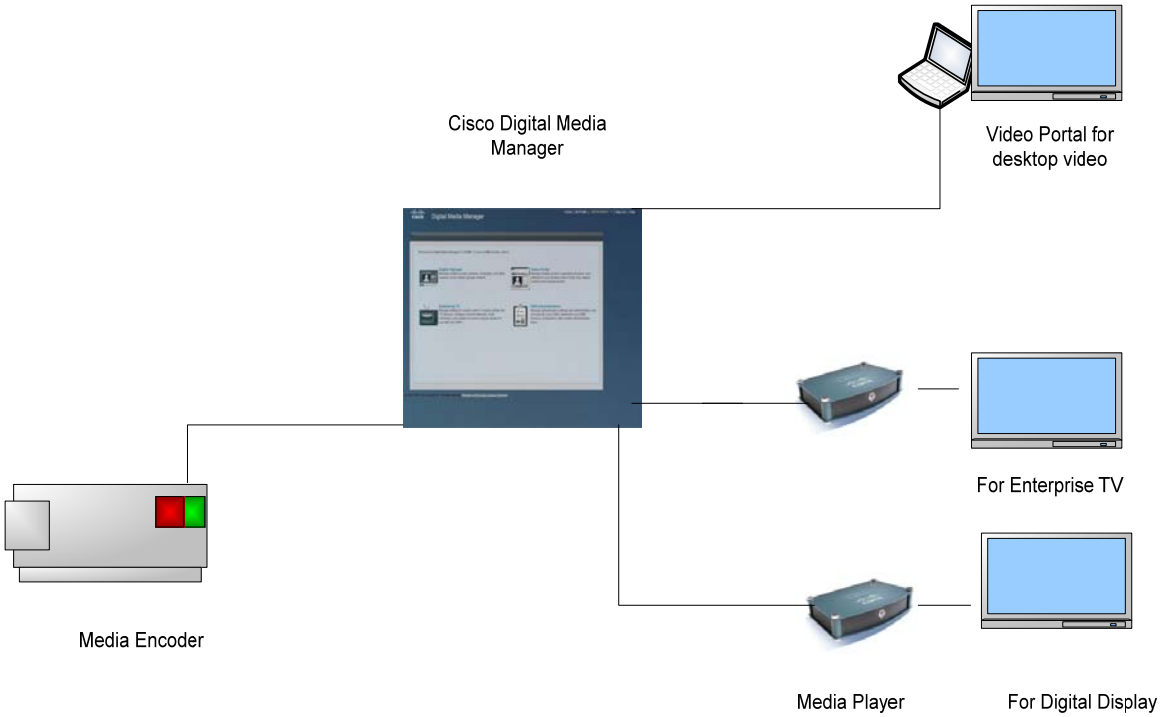

Figure 4-1: Cisco Digital Media System [1]. (Appears with the permission of Cisco)

4.2 CAYIN Technology Network Architecture

CAYIN deployed web based digital signage is based on two network architectures [18]:

- Networked Stand-alone Digital Signage
- Digital Signage Network with a Server-Client Structure

4.2.1 Networked Stand-alone Digital Signage

Networked stand-alone digital signage is an open structure deployment that can be easily integrated with a customer's existing network infrastructure (see Figure 4-2).

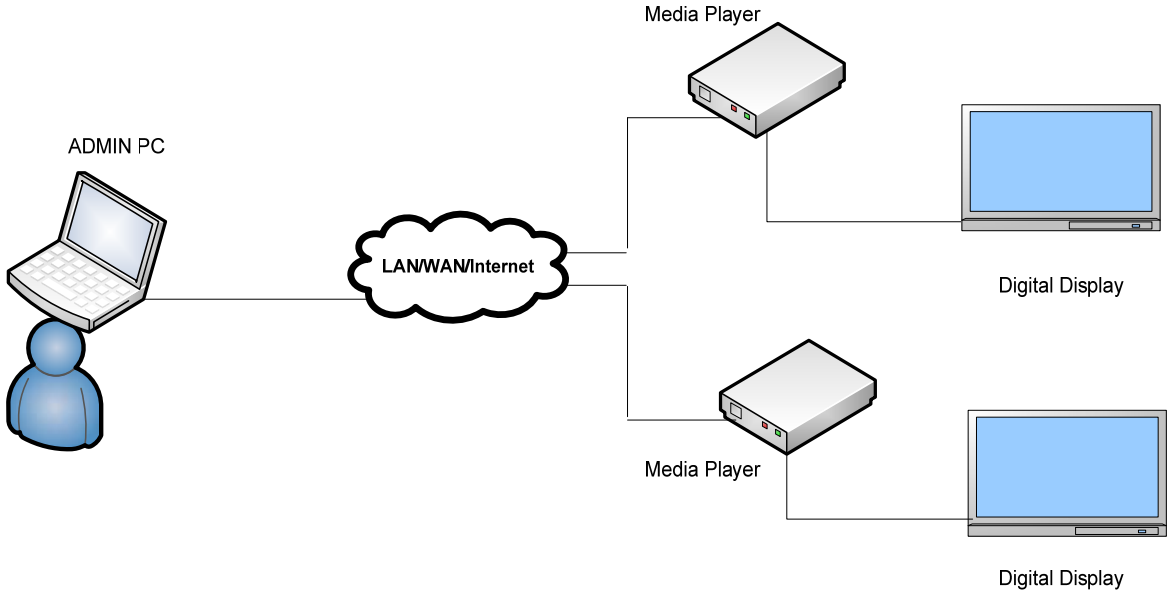

Figure 4-2: Networked Standalone Digital Signage

4.2.2 Digital Signage Network with a Client-Server Structure

Alternatively, a digital signage network can utilize client-server architecture (as shown in Figure 4-3). This approach can also be easily integrated with a customer's existing network infrastructure. Note that in this approach the media players act as clients to a network attached media server.

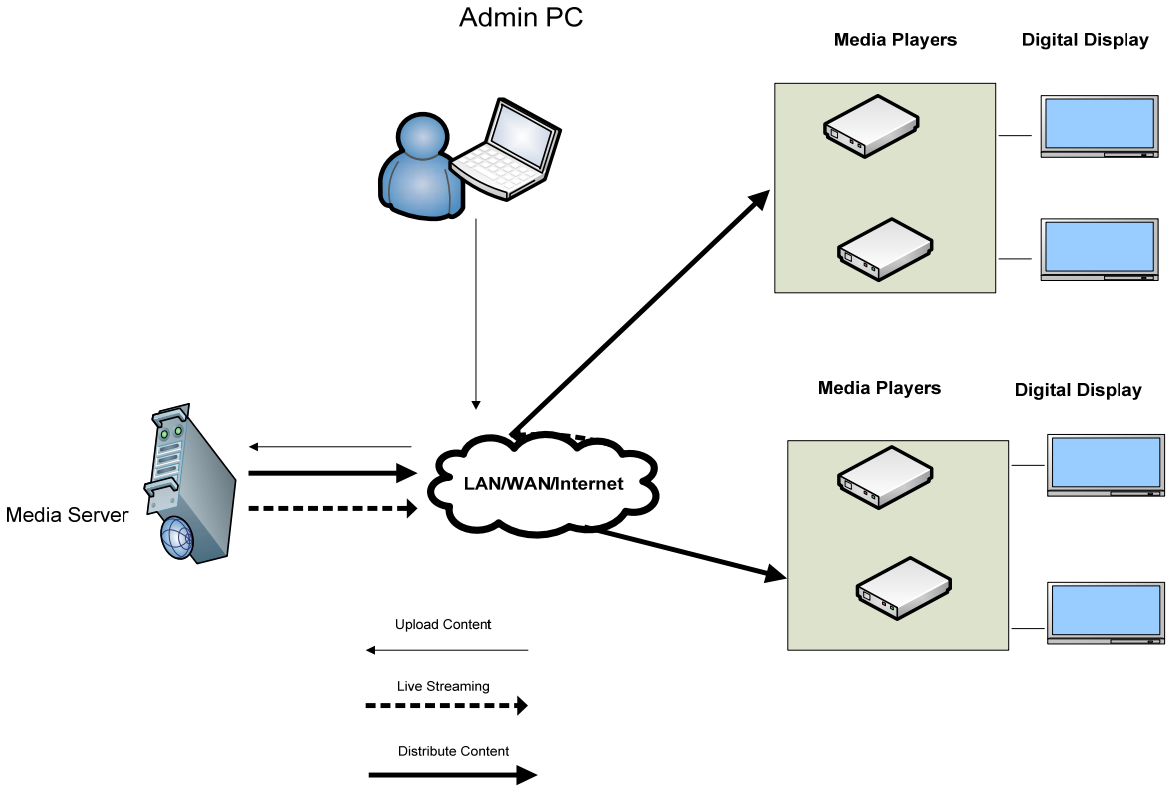

Figure 4-3: Digital Signage Network with a Server-Client Structure

4.3 Summary of network architectures for digital signage

Table 2-1 summarizes our observations about the two major architectures for digital signage. We note that client-server architecture is available from both of these companies and is based upon streaming media. CAYIN Technology also supports a managed standalone approach which is close to the web (browser) based digital signage approach in which the contents of the signage/display can be directly controlled by a local web browser.

Table 4-1 : Summary of Network Architectures for digital signage

Company/Organization	Network Architecture
Cisco	Based on WAE and Cisco Application and Content networking Application software to distribute and control contents.
CAYIN technology	**Networked Standalone Digital Signage** • Web interface, content can be managed by web browser. **Digital Signage Network with a Server-Client Structure** • Based on Media server • Server is responsible to manage and distribute the contents.

Chapter 5 - Market Analysis

Digital Signage refers to electronically controlled display of messages/information that can be updated without the cost and expense of changing a physical sign itself [19]. For retailers and business organizations digital signage may be cost effective and can increase revenue and improve customer satisfaction. The digital nature of the system allows contents to be changed quickly and cost effectively. For educational and research organizations, valuable information & messages can be delivered to targeted viewers at a specific time in order to improve communication between the organization and its customers and employees.

Digital media plays an important role for retailers, advertisers, and consumers. According to [20]:

- 70% of consumers make purchase decisions after they arrive at an outlet.
- 75% of consumers say the point of purchase influences their decisions to purchase.

They go on to state that digital media is:

- 5-10 times more likely to be noticed if it uses dynamic rather than static media.
- Customers are 2-5 times more likely to recall dynamic contents rather than static contents.

Point of sale and other advertising has historically used print and traditional static signage to communicate marketing messages to shoppers and consumers [20]. As noted earlier, static/conventional signage has many disadvantages in terms of its construction, cost, distribution, and installation. However, the main disadvantage of traditional signage system are that it is time consuming and expensive to change the contents. The cost of material and human resource leads to high cost and the requirement for a substantial budget. Advertisers and marketers are looking to digital signage for greater revenue and more effective communication. According to [21], industry experts predict that dynamic digital signage will reach US$ 3.4 billion by 2009.

If digital signage is adopted by an organization, then the contents must be high quality, updated frequently, easy to understand, relevant, and timely. Such digital signage can help to educate customers and encourage them to make a purchase decision or facilitate other actions by the viewer.

Chapter 6 - Related Work

A pervasive display system is a combination of multiple public displays. These public displays need not be under the control of single user. The main function of the system is to manage display requests and display resources. Creation of such a system supports coordination between multiple displays, even if some displays are dispersed over a (possibly remote) geographic area. We will assume that each user can interact with only one display at any point in time; although this point of interaction can change as the user moves about.

Rui Jose categorized pervasive display systems into different models depending upon user and organization requirements. Some of these application models are [22]:

Experience Oriented These are part of media and sensor rich installation displays. They have very strong interactions with users. Typical examples are games and interactive art installations.

Content Oriented Content oriented displays are focused on displaying content to people passing by. The contents depend on the people and time. These kinds of digital displays can be easily controlled by a digital signage network system.

Sign Oriented Sign oriented displays are used as digital replacements for traditional signs for some nearby event, the name of a place, or other directional information.

Ambient Oriented The main purpose of these kinds of displays is to deliver *locally* interesting information. Examples may include GUI-Based approaches such as informative art [23], InfoCanvas [24], and displays based on light & sound patterns as in the Hellow.Wall [25] and Ambient Room [26].

Personal Oriented Personal oriented displays are mainly designed to support individual access to digital services. They differ from other displays in that they are located a public place, but they provide specific information to specific people. Because some of the user's information will be private this information should not shown on the display if there are other people present. Typical examples of personal oriented displays include store assistance displays in which customers may approach the display with the barcode of a product visible to a barcode reader and obtain further information about the product. Other examples are Dynamo [27] and bluebeards [28].

In [29], John V. Harrison and Anna Andrusiewicz compare the deployment of traditional/conventional signage and digital signage systems in terms of cost, time, distribution, and installations. They show that compared to digital signage, the "Creation, Distribution, and Installation" process of conventional signage is expensive. They examine this process for both technologies in informal process diagrams (see Figure 6-1). They explain that by implementing a digital signage network many steps will be eliminated as compared to a traditional signage system. The vertical dotted lines show the workflow to

deploy a conventional signage system while the solid arc illustrates the process when using digital signage.

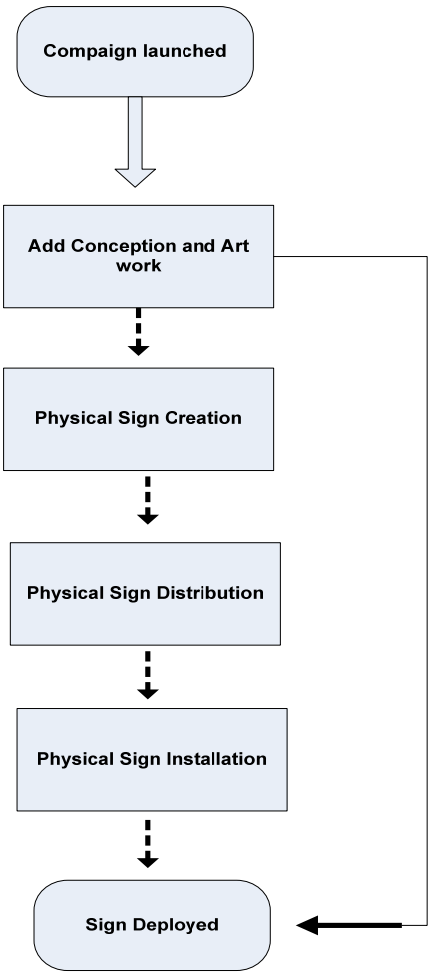

Figure 6-1: Eliminating the creation, distribution, and Installation cycle via digital advertising and a digital signage network [29]

They also proposed a digital signage exchange system which an organization that owns a digital signage network can use for their in-house advertising requirements, and could also sell multimedia display time to advertising agencies or trade display time with other digital signage network owners.

In [30], these same two authors describe the importance of narrowcasting multimedia advertisements and other messages, using a digital signage network. They proposed a model of how a viewer or users can interact more with digital signage through wireless access points. As a result of this interaction it is possible to gather interesting and valuable information about the advertisement of products. Viewers who are interested in obtaining additional information about the products displayed via digital signage can interact using their wireless device and indicate that they wish to receive more information. In response the display controller will transmit documents related to the product to the user's wireless device. If the information about the product is extensive, then the display controller simply transmits a URL to the user. If the user wishes, then he or she can send information via the display controller to the advertiser over the digital signage network. Based on this information the user allows an advertiser to directly contact them in order to provide more information about the product or perhaps, in the case of digital products, to directly purchase and receive the digital product.

In [31], these same two authors presented the further details of their concept for a digital signage exchange. They have implemented a prototype of such an exchange. As stated earlier the idea is that a firm can use their own displays for in-house advertising/information as well as sell display time to advertising agencies or trade display time with others. This allows multiple organizations to form a digital market via which advertisers can purchase display time. In this paper they developed the concept of a transaction management model for this exchange. This model describes a formal representation of a business order for display time. Each order consists of the contents to be displayed along with a content ID, exposure duration, activate time, expire time, cycle duration, exposure frequency, cycle gap, and schedule ID. The output of the transaction management system is a set of display schedules for all of the display controllers. These schedules are distributed via the digital signage network.

Jörg Müller, et al. [32] present digital signs from a very different and interesting aspect – in that they propose the use of digital signs that automatically learns the audience's preferences for certain content in specific contexts, then the digital sign presents these contends accordingly. Cameras at the digital signs observe the audience and detect if someone is watching the content via face detection. The viewing time of the content can be stored in a database together with date, time, and the sign's location.

When scheduling content, each sign calculates the expected viewing time for each content type depending on the sign's location and the time of day using a Naive Bayes Classifier. Based upon this the weighted viewing time of contents are selected randomly. They have implemented this system in a university on four digital signs. They collected some measurement over a period of two months about the audience reaction to contents. Their system uses a face detection algorithm, a MYSQL database, a Java based scheduler, and a Java based player. For the face detection they have used special software called Fraunhofer IIS to analyze the video stream [32] [33].

After implementing this system they conducted interviews within their department to learn the perception of people regarding the performance, effect, and other issues related to this system. The variance of the view time indicated that the influence of the chosen content categories on view time was relatively small. Another surprising and important result was that there was no correlation between viewing time and the user liking the content. Thus users may even look at contents that they do **not** like. Some users objected to the use of a camera, some users even thought it was acceptable, and some others did not care.

In [34], Jörg Müller, et al. showed some interesting aspects of the audience *expectations* towards what is presented on public displays; specifically that their attention is correlated with their interests. This is similar to the effect of "Banner Blindness" on the web, in that users often ignore contents (e.g advertisements) that they find uninteresting. Further investigations have studied this issue in two phases. In the first phase they interviewed 91 users concerning 11 different public displays. Most of these displays were placed at different locations at the university and in shop windows. One display was fixed to a public telephone. Three were fixing in a Café and hotels to display video programs. Interviews were conducted regarding these displays. There were a variety of different opinions; some showed interest while others show uninterested. Based upon these interviews two different major factors seemed to affect whether participants looked at public displays. A comparison between the university displays and other displays showed that displays where participants expected something interesting (for them) the displayed content received a lot of attention in various locations. However, displays where the

participants expected nothing interesting were largely ignored. Another interesting observation was that some people wanted to see very different content, expecting personalized contents.

In the second study phase they conducted interviews with 17 people to identify the dimensions that users believe to affect whether they look at public displays. They chose a team of different age people to conduct the interviews. They compared the different dimensions of the digital displays with respect to the ratings by the participants. These dimensions were categorized using affinity analysis [34][35] and for each category a mean correlation with the rating of whether participants believe they would look at a display were computed. They have concluded that similar to the effect of "Banner Blindness" there is an effect of "Display Blindness" meaning that uninteresting and unexpected contents lead to ignored displays. They proposed that in order to overcome the problem of "Display Blindness" display owners should investigate audience expectations for certain displays and then the content should be designed to fit these expectations.

Jörg Müller, et al. also presented some advertising techniques and approaches that automatically optimize digital signage[36]. The idea is that the advertisements slots (a timer period that a display could be used) are sold in an auction. Each advertisement is represented by an agent who is given the context of the display. From this context, the agent determines how much to bid in auctions. The agent who wins the auction gets to schedule their content for the display at a specific location and for a specific time period. The content is presented along with a visual coupon. Customers can take a photo of that advertisement with their mobile phone and use this photo as coupon to show to a cashier. The coupon consists of a code which contains information about when and where the user has seen the advertisement. This code provides feedback into the system and can be used to measure the effectiveness of an advertisement. Due to this feedback, advertisements which are liked more are shown more frequently and on more displays.

Chapter 7 - Design and Implementation

This chapter describes the objective and goal of our prototype (of a browser based digital signage system). We will begin by first stating our method and goal; then go step by step through our design and implementation phases. The next chapter will discuss in detail how we developed our system.

7.1 Goals & Methods

As stated in section 1.2 on page 2, the main focus and objective of this thesis was to design, develop, and evaluate a hardware/software solution based on a PC, to control and display different web contents (or other dynamic information) on digital displays. Instances of this system could be located at different locations. A user or administrator will specify some content or dynamic information that is to be displayed on a certain digital display for a specified time and for a specific period of time.

The approach that has been selected is to follow the web based approach described in Section 2.3. This approach was selected to meet our secondary goal of removing the single point of failure of a television (be it analog or IPTV) based system. This required that each display has an attached processor and local storage containing both the information to be displayed and the local playlist of what is to be displayed. The contents of the signage/display can be directly controlled by a local web browser. Figure 7-1 shows the main elements of such a web based digital signage system.

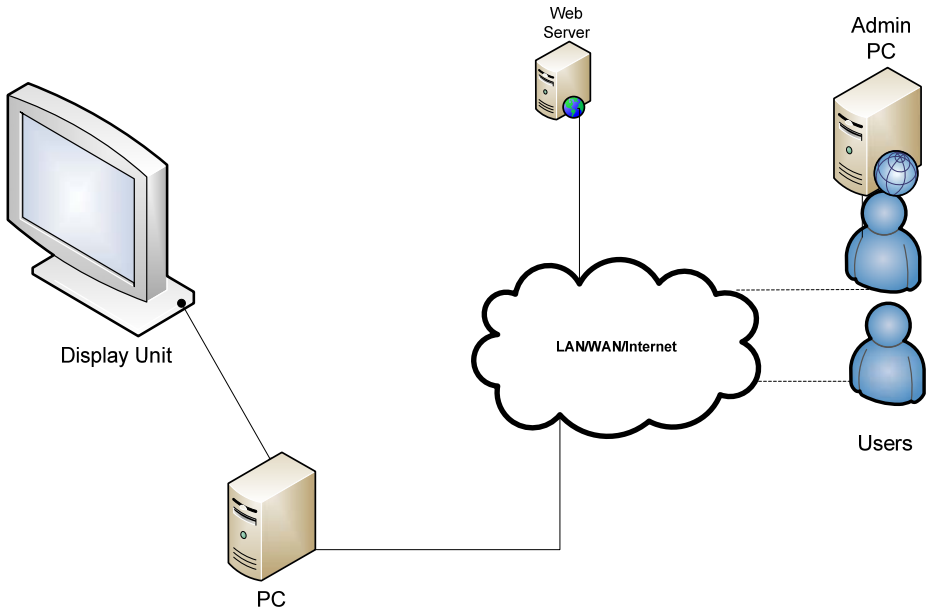

Figure 7-1: Web-Based Digital Signage

While the PC could be connected to one or more LCD displays at different locations, for simplicity we will begin by assuming that each such PC controls a single display as shown in Figure 7-2. We have chosen to use a PC based approach as there are a very wide range of PC platforms available to use – enabling an organization to makes choices according to their own requirements (in terms of cost, availability, performance, power, cooling (if any), video (and possibly audio) interfaces, etc.).

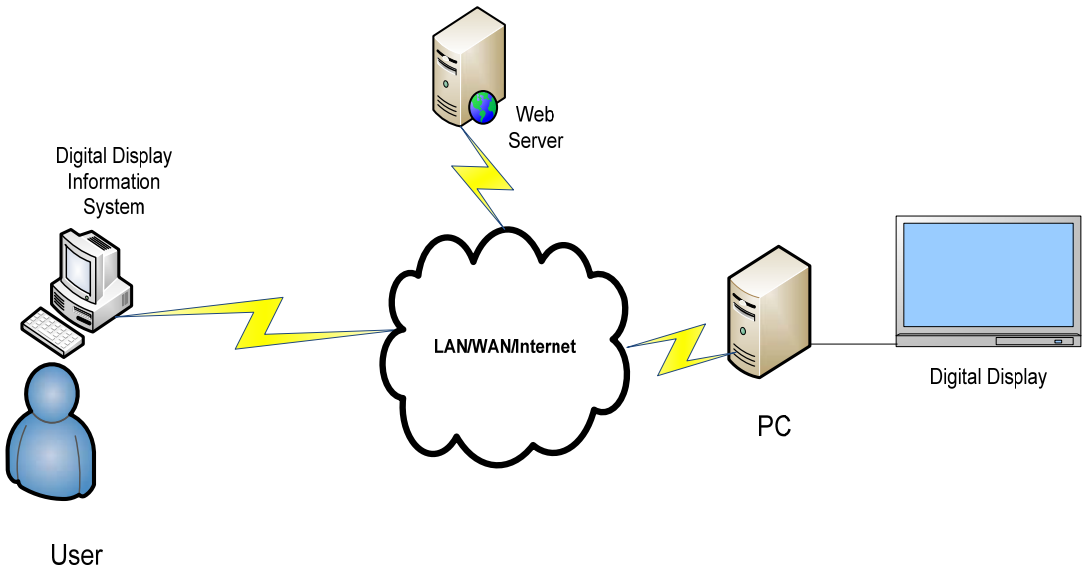

Figure 7-2: Browser based solution with a centralized control system

The centralized management system is implemented as a web service; hence it can be accessed through a web browser. Based on access controls, dynamic web contents can be sent to different digital displays based on the list of pages assigned to each of these clients (i.e., the PCs with web browsers connected to displays). An administrator or other authorized employee can specify the schedule (a playlist), i.e., a list of URLs or web dynamic information to be displayed by specific displays along with the specified schedules, e.g date, time, and duration of the content information on each display. The digital display management software can manage the different schedules for each display.

In the implementation of our system, we needed a user friendly interface to allow the employee to input this information. As will be described in section 7.3.2, we based our digital display and management software on an open source software package called MRBS (Room Booking System). This software fulfills all of our requirements and required only minor changes to adapt it for our own purposes.

We also needed two other applications SOAP (Simple Object Access Protocol) server and SOAP client. SOAP server to retrieve all the relevant information from the database and download the content to the client machine. (See section 8.2 on page 36 and Figure 8-12 on page 42.) Each client (PC attached with display) has a SOAP client and a web server. The SOAP client will take all the requested data from the web service and will store it in a XML file. Later this saved XML data will be used to schedule the output to be shown on display.

7.2 Lab Environment

In order to implement a digital display management system, we need both hardware and software. In the next subsections we will give a detailed description of this hardware. Following this we will describe the software running on each machine, including its dependence on the underlying machine.

Figure 7-2 clearly shows that we need at least two computers and one digital display. The first computer will be used as a server to execute the digital display management software (both the web service and a database containing content and schedules). For this we are using an existing server machine (a Dell PowerEdge 2850 rack mounted server equipped with dual Intel Xeon processors at 2.8GHz, 2 Gbytes of memory, LSI Logic

53C1030 Ultra 320 SCSI disk controller, dual 36GB disk (configured as a RAID), dual Intel 82541EI Gigabit Etherenet interfaces, and dual power supplies). For web server we will use the same server machine.

The client (PC attached to display) we used a simple laptop (Dell model D600 with 1GB of RAM, 1 GHZ processor, and 20 GB hard disk). As a display we used a Sharp AQUOS 32 inch large screen TV. This large screen TV display was already installed at Wireless@KTH. The display is connected via a VGA cable and the display is set to 1280 by 1024 @ 60Hz display model. Additionally, we connected a USB to serial adapter to this computer and connected to the serial input of the large screen TV in order to be able to send commands to the display (for example to turn the display off at the end of the day).

The server and client machine were connected via the Wireless@KTH lab's internal Ethernet network. While the network was capable of supporting 10/100/1000 Mbps Ethernet connections, the 10 Mbps interface of the laptop limited the speed of the communications between these two machines. It should be noted that as we did not use streaming video – the network was not a bottleneck in any of our testing.

7.3 Software

7.3.1 Operating Systems

In order to build a stable system environment and not be bother by virus infections, Linux was chosen as the operating system for the server (OpenSuse 10.3, with a Linux 2.6.22.12-0.1-default kernel) and while the client ran Microsoft Windows XP. Note that because the only software needed on the client side is: (1) a web browser, (2) a SOAP client to download content and playlist to the client, and (3) a custom program to send commands over the USB attached serial port to the display. Hence the client could just as easily have been running Linux (in fact, PCs running Linux were also tested with this display).

7.3.2 Application Software

As was mentioned above, we based our digital display and management software on an open source software package called MRBS (Room Booking System)[37]. MRBS is a web application written in PHP and MySQL for booking meeting rooms, but here we used it to book a digital sign's display slot to display different contents.

We used MRBS software because our requirements are exactly same for which purpose this software is designed except that MRBS was designed for room booking and we used it to schedule the display contents on a digital display.

We used the MRBS software as our base,for the following reasons:

- It is easy to use and is both simple and reliable.
- It is open source
- It has a user friendly web interface.
- It fulfilled most of our requirements for the server -side user interface software -- which saved us a lot of time during the implementation phase.
- This software was already used by the Wireless@KTH staff for room booking, so most of the staff is aware of how to use this software – reducing the time and effort that might be need to train the staff to use the system.

We needed to make some changes in the MRBS software to meet our requirements; such as replacing a room by a display. We will discuss some of the other important changes latter. We will also refer to this software as a "Digital Display Information System" instead of a Room Booking System in the remainder of the thesis.

We began by downloading and installing the MRBS software. It can easily be downloaded from its website [37]. There are some prerequisite for installing this software. Apache is needed to provide a web server. MySQL is needed to store data. PHP is needed to utilize dynamic web pages. Both Apache and PHP are open source software which can be downloaded from their official websites ([38] and [39] respectively). MySQL has been downloaded from its official website [40]. We downloaded all these packages from their official websites and installed/configured them using their support documents.

Next PHPMyAdmin was installed to administer the MySQL database via a web interface. PHPMyAdmin is a free software tool written in PHP intended to facilitate the administration of MySQL over the World Wide Web [41]. PHPMyAdmin supports a wide range of operations with MySQL. The most frequently used operations are supported by the user interface (managing databases, tables, fields, relations, indexes, users, permissions, etc), while you still have the ability to directly execute any SQL statement. Managing privileges, keys on fields, and export data into many formats. The major advantage of PHPMyAdmin is to make the administration of MySQL database much easier through using its graphical user interface. The PHPMyAdmin interface is shown in Figure 7-3.

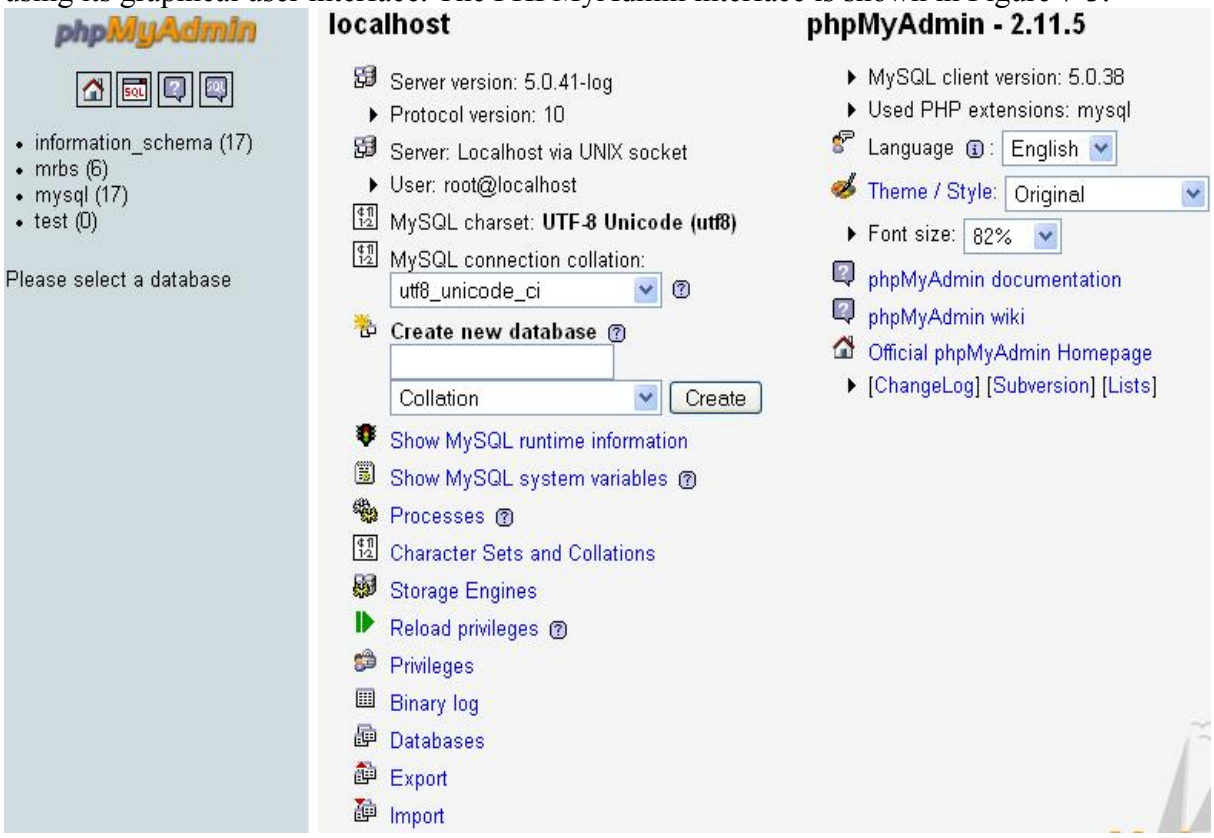

Figure 7-3: PHP MyAdmin Interface

In the above figure the left column shows the name of the databases currently in the MySQL database. The middle column shows some general information about MySQL, including the server version, protocol version, and user information on top. In the middle section it shows information needed to create databases and to perform some other

operations. The right columns give us information about the program and its documentations.

After the installation of Apache+PHP+PHPMyAdmin, we now had to configure MRBS following the instructions in the MRBS "INSTALL" file. We placed MRBS in the directory /var/www/mrbs. After configuration of MRBS we configured its database using PHPMyAdmin. We created all its tables using PHPMyAdmin, specifically mrbs_area, mrbs_display, mrbs_entry, and mrbs_repeate. As noted in the beginning of this section we replace the room by a display. This is because we are going to book a display for a specific time to display specific information, similar to how a normal room booking. We defined a number of areas into the table mrbs_area (specifically Wireless@KTH, Forum main building 8th floor, the Software Engineering department, and the KTH library in Kista. We added entries for a number of different displays into the mrbs_room (display) table (such as a main display, teacher display, etc.). After doing this the basic Digital Display Information System configuration was finished.

The resulting database structure is shown in Figure 7-4. This database structure contains four database tables. The table mrbs_area represents and manages different areas where different types of displays can be added to each area. The table mrbs_display stores information about the different displays (such as their IDs, names, and locations). The table mrbs_entry contains entries with an entryid, start time, end time, the description of the contents, content type, and pointer to the contents. The last table mrbs_repeat stores the repeat time and date for when the contents will be displayed.

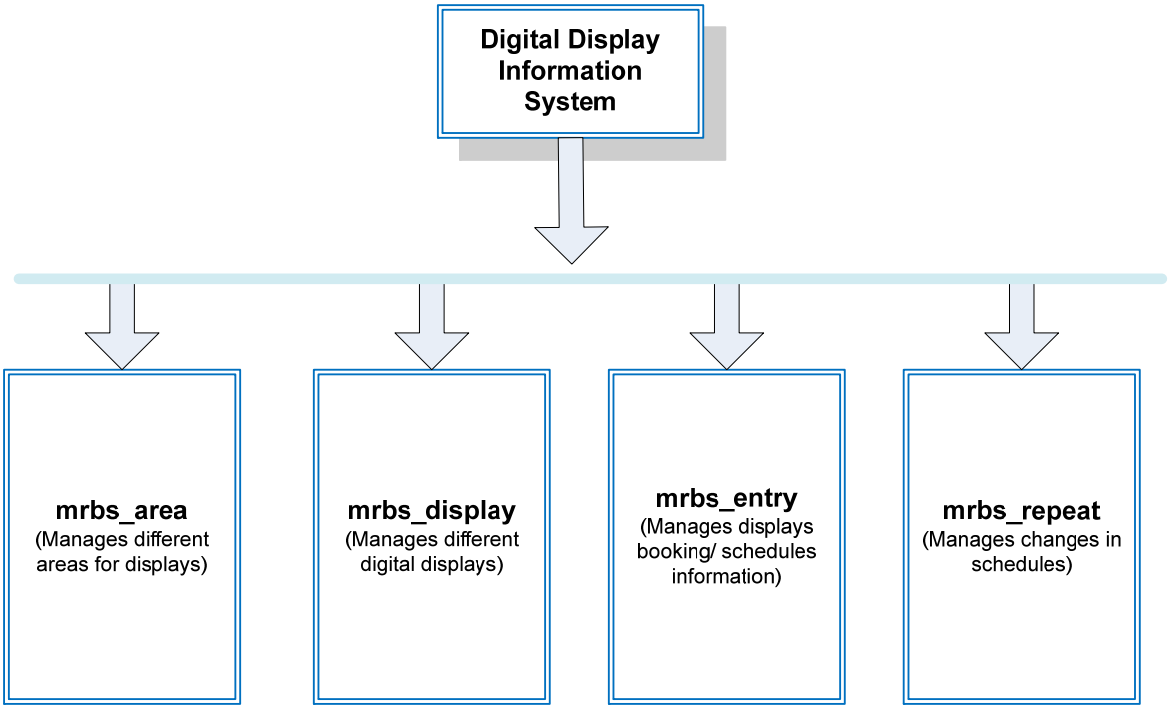

Figure 7-4: Database table structure of the Digital Display Information System

The server host name, database name, and login username and password were defined in the configuration file of mrbs to enable PHP scripts to connect to the database. In addition, we made some other changes in the configuration file of MRBS, such as changing the name of the software from "MRBS" to "Digital Display Information system" and set the name of the company to "KTH, Royal Institute of Technology, Wireless@KTH". Now

everything is ready to run this digital display information system software. Figure 7-5 shows the main home page of the digital display information system.

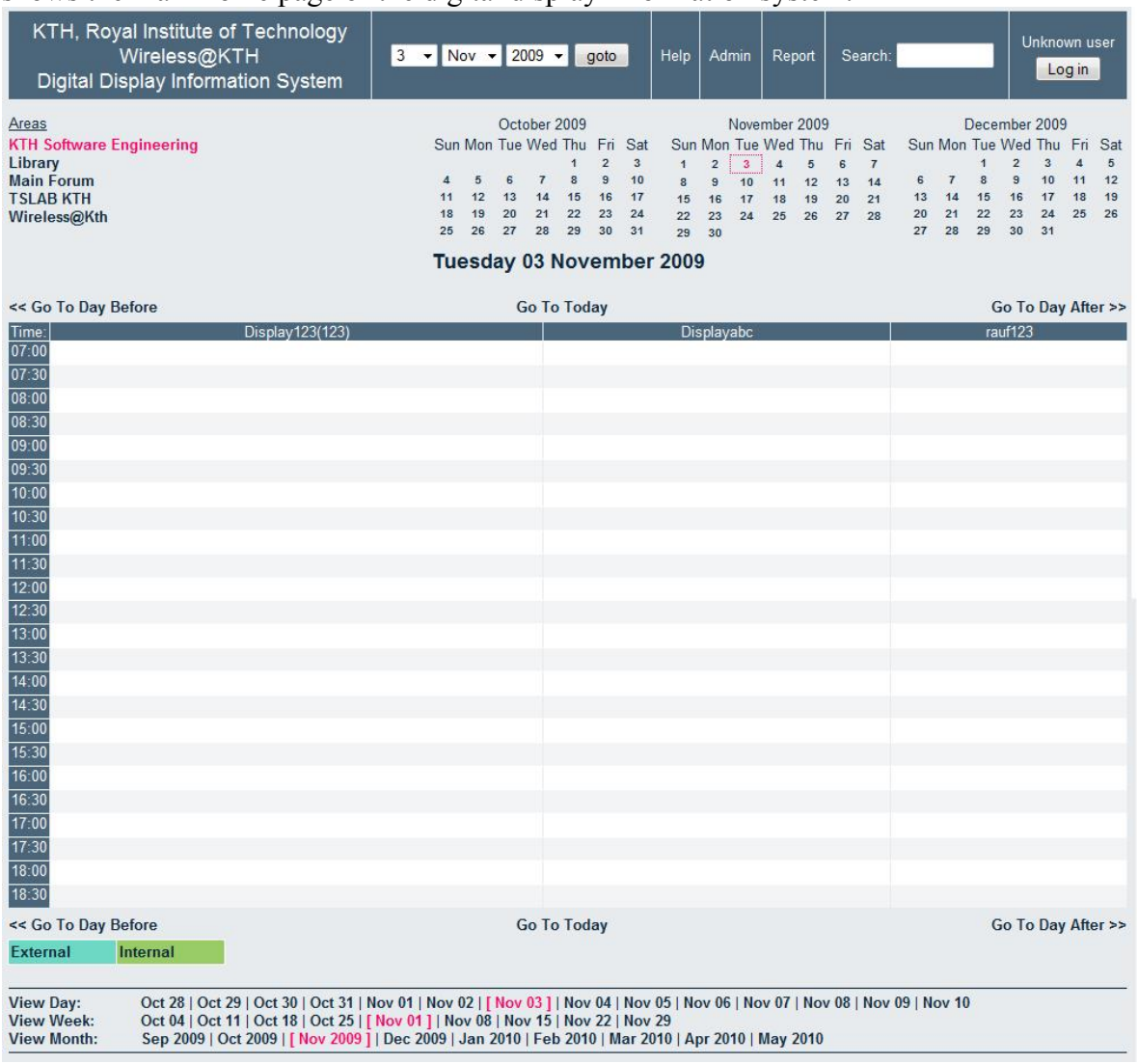

Figure 7-5: Main Interface of Digital Display Information System

The header section of this page shows the name (title) of this software, the identity of the current user, date, time, and some other information. Below the header there is a list of areas and calendars. If we select the first area, e.g 8th floor, all the digital displays will appear below in the middle of this page. Only the administrator has the permission to add digital displays, whenever a new one is installed, etc. A time schedule and other information such as types of contents, URLs, images, and descriptions will be shown under the specific display.

Any user can click on the whitespace under a specific display, to display another page called “edit_entry.php”. Here the user can enter the information about new contents to be displayed. (See Figure 7-6). If the user wants to display the page as a URL, the user will select the “link” radio button, and then enter the URL. Alternatively, the user can specify a file (for example a JPEG image) by select the “file” radio button, then uploading this specific file. Other fields on this form include: date, time, duration, area, and the selection of the digital display where a user want to display content.

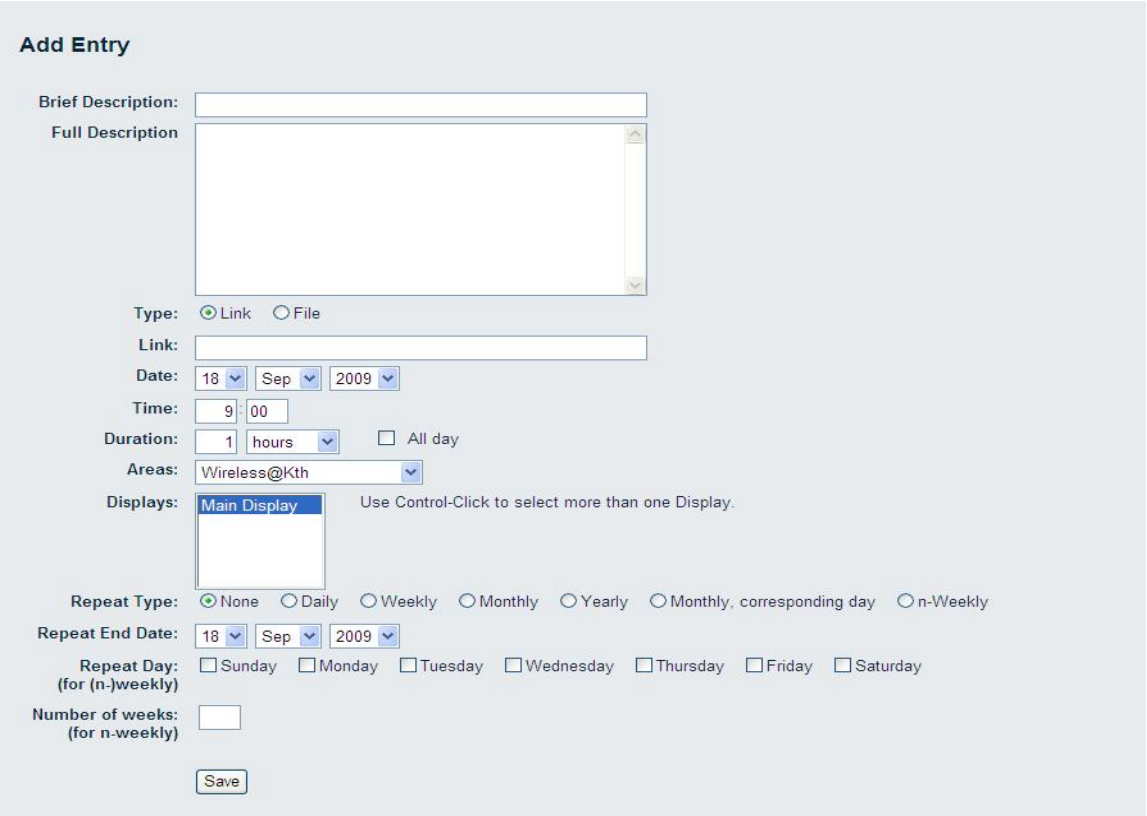

Figure 7-6: Add Entry Form

Chapter 8 - Prototype design

Designing the prototype of a digital signage system was divided into four steps. The first was to modify and add several modules in the MRBS system. The second was to create a web service to convert our application into a web-application via a SOAP server and SOAP client. The SOAP server will query the database to get all the necessary information for the digital display information system. The SOAP client on the client side will make a request to get all of the relevant data (both contents and schedules) from the SOAP server, in order to store this information locally. The third step is to display all the contents according to their specified schedule on the display attached to the client. The last step was to test the implementation at Wireless@KTH to show that it work with this specific display as expected.

8.1 Modifying & adding modules in MRBS

Since the MRBS software was designed to be a room booking system we made certain changes to meet our requirements of a display booking system. Fortunately, most of our requirements were already fulfilled by the MRBS software. As we noted earlier, our first change was to replace the room by a display (as we are booking display and not rooms). We also made some other changes as MRBS did not have support for uploading files or selecting a link. Finally, we also made changes to enable the user to specify multiple elements to be displayed. Rather than resulting in a conflict when multiple items are scheduled during the same time period, these items are simply displayed in consecutive time slots. These items will continue to be displayed in a cycle for the scheduled period.

8.1.1 User Login and Authentication

To access the MRBS software we need a valid username and password. The login page is shown in Figure 8-1. We cannot perform any actions until we have supplied a valid username and password. By default the configurations of the MRBS software gives us the facility for setting several different usernames and passwords – in addition to specifying different access control mechanisms. In the configuration file (config.inc.php) of MRBS in the section “Authentication Settings” you can specify multiple username and password pairs. These are specified with the following format:

```
$auth["user"]["rauf"] = "rauf";

$auth["user"]["chip"] = "chip";
```

Figure 8-1: Login Page

Initially, we set an administrator password. The administrator has full privileges within this system, while a normal user has only limited access. When a user is logged in as an administrator he or she can specify a new area, add a new display, etc. while other users cannot perform these operations. The administrator must also specify valid usernames and passwords for the other users, although they can specify that this is done via other mechanisms, such as the user's Kerberos IDs, database based authentication, LDAP, IMAP, POP3, etc.

8.1.2 File Uploading and Link Selection

The MRBS software did not support a file uploading option to be used for uploading files, such as images or PDF files that we might want to show on a digital display. Therefore we added a file upload option that saves the file in a specific directory. Later this content will be sent to a client to show on the digital display. In our initial implementation this content will not be sent to the client until the client sends a request for this specific file. This file uploading option was added to the edit_entry.php form. It is invoked when a user selects the type "File" by clicking on a radio button. The user can upload any file he or she wants to display. The file will be saved in the directory /var/www/mrbs/web/Files. Figure 8-2 shows this portion of the web page. (See Appendix A for the modified source code.)

Figure 8-2: File Uploading

We made change in the database by adding fields in the mrbs_entry table. To support file uploading we have added a field to this table to record the file information and its type. We have to record whether it is a link or file, and if it is a file then the file will be uploaded on the server in the specified directory (as mentioned in the above paragraph), otherwise a URL (link) will be stored in the database.

Similarly if the user wants to specify a link, then they click the radio button and write the URL in the text box – as shown in Figure 8-3. (See Appendix A for the modified source code).

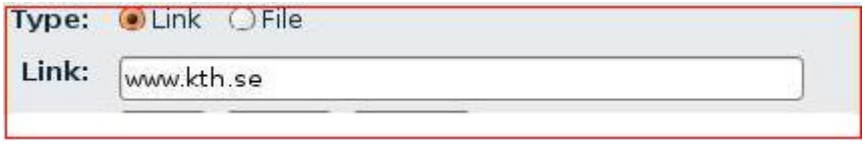

Figure 8-3: Link Selection

8.1.3 Multiple Entries on the main page of MRBS

Initially, MRBS did not support multiple entries on the page "day.php" because; it assumed that there would only be one entry per room at each time. This is because it supports exclusive booking of a room at a specific time and for a specific time period. Because it enforces this exclusion, there cannot be overlapping or multiple bookings for a room. Attempting to book such a overlapping or multiple booking will cause the software to output a message indicating a conflict in bookings. Therefore, we modified the software to support multiple entries for a block of time. Figure 8-4 shows multiple entries for the time interval 9:00 to 9:30. In this case the URLs and files will be shown in a cycle on the display for this half hour period. (See Appendix A for modifications to the source code).

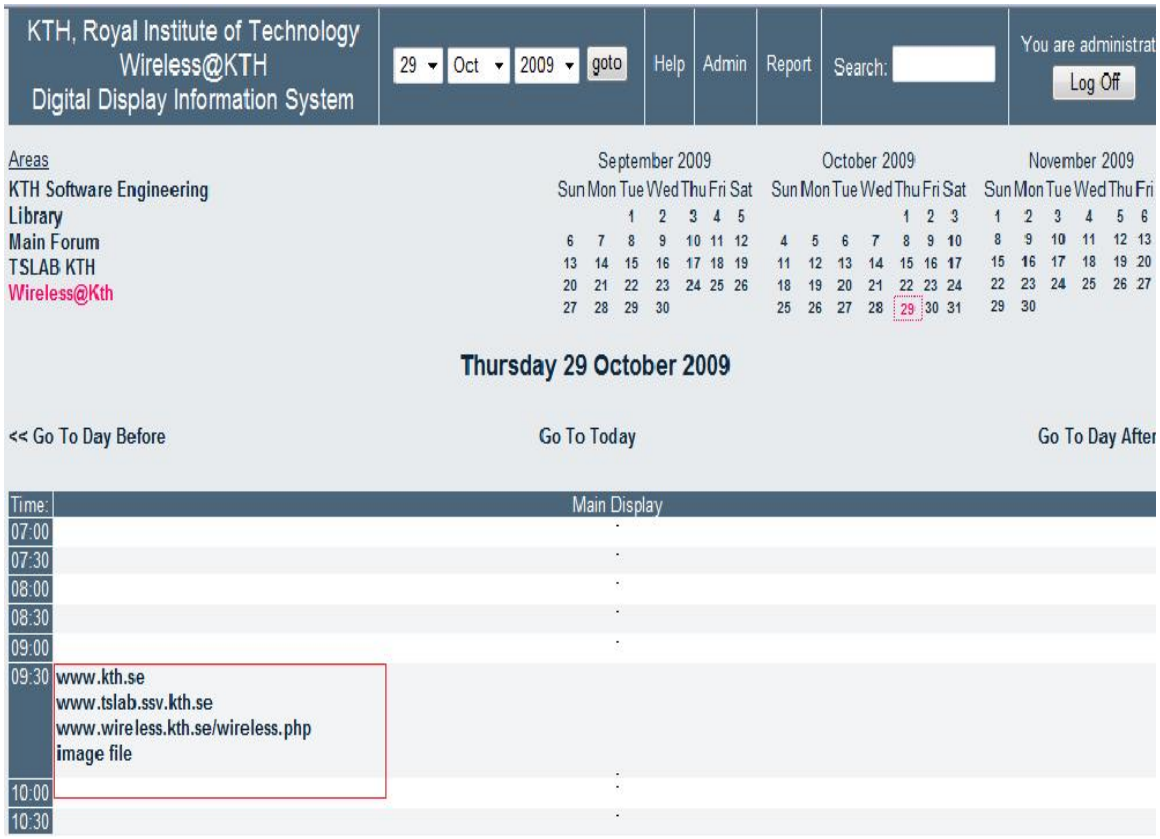

Figure 8-4: Multiple Entries in the main page of Digital Display Information System

8.2 SOAP based Web Service

Next we created a web service that will exchange messages through SOAP between a client and a server. This will enable the client to retrieve all the dynamic information it needs. This dynamic information includes the schedule and the contents to be shown according to the schedule.

Web Services are application components that communicate using open protocols; such as XML and HTTP [42]. XML is used to transport and store the data; while HTTP is used to exchange the information between the client and server. We implemented a web service for the following reasons:

- Web services communicate using open protocols; thus making it easily to add other clients and servers to the system.
- A web service supports all kinds of web browsers and many different platforms.

Web services depend on the following platform elements:

- Simple Object Access Protocol (SOAP)
- Web Services Description Languages (WSDL)

The Simple Object Access Protocol (SOAP) is a communication protocol for exchanging structured information to implement web services [43]. SOAP is based on XML and can exchange information over HTTP. Through SOAP we can easily access our web service. We have used SOAP protocol for the following reasons:

- SOAP is platform independent.
- SOAP is simple and extensible.
- SOAP is language independent.
- SOAP communicates over HTTP, and HTTP was designed to support all internet browsers and servers.
- SOAP provides a way to communicate between applications written in different programming languages and running on different operating systems.

The Web Services Description Languages (WSDL) is an XML based language for describing web services and how to access them 0. WSDL describes the web service, including its location and the operations the service exposes.

8.2.1 Creating a Web service

To develop our web service we selected the NuSOAP toolkit for PHP. It is a set of PHP classes; hence no PHP extension is required. This allows developers to create and consume web services based on SOAP 1.1, WSDL 1.1, and HTTP1.01 [43]. NuSOAP can be downloaded from its sourceforge.net website [45] and placed in the folder of webservices (on the server this is the directory /var/www/webservices). The toolkit contains two folders: Samples (containing sample clients and WSDL client files) and lib (containing the libraries; classes for SOAP, WSDL, SOAP clients; and a SOAP server). We do not need to change anything in these files. Browsing to http://localhost/webservices/ will show the contents of the webservices directory. Only the two folders lib and samples can be shown, as Figure 8-5.

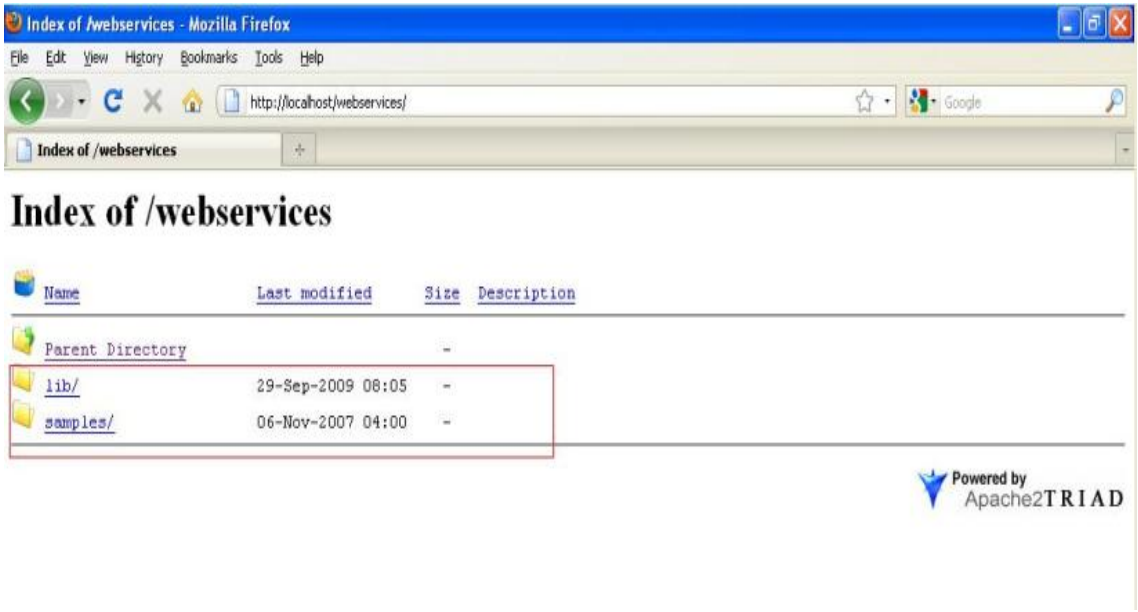

Figure 8-5: Web services Directory Contents

Next we have to launch the web service by creating a SOAP server. First we have to load the SOAP libraries that implement the functions we will be making available via the web service, then we instantiate a SOAP server to receive and respond to the SOAP client's request(s).

First we have to create a SOAP server by calling "`new soap_server`". Next we configure a WSDL file which can contain multiple services. This WSDL file should describe the web service, specifically giving the names, and types of input and output parameters. We also tell the NuSOAP server that we want to support WSDL. We have created a function 'information' which contains the implementation of the web service and receives the SOAP inputs directly as function parameters. Finally, we register this function with the web service to expose its input and output operations. All of these steps are shown in the following source code:

```
require_once 'lib/nusoap.php';

  $soap = new soap_server;
  $ns = "http://www.kth.se";
  $soap->configureWSDL('Digital Display Information System', $ns);
  $soap->wsdl->schemaTargetNamespace = $ns.'/xsd/';
  $soap->register(
    'information',
    array('a' => 'xsd:int',
    array('b' => 'xsd:Array'),
    $ns
  );
```

The web service is now created, but we also need some additional information – the content which will be requested by the SOAP client and passed to the web service, e.g display_id, start time, end time, link, file, etc. This information will be returned by the SOAP server after querying the database using the display_id. The web service/ SOAP server will send this information in a response to the SOAP client. The information is returned in an array as specified by the definition of the output about (i.e, the definition of the outputs in the line "array ('b' => 'xad:Array'). The line just above this says that the input to the function will be one integer (i.e the display_id). The (PHP) source code below,

shows that the elements of the array are each a set of records, with each record having an ‘id’, ‘start_time’, ‘end_time’, ‘name’, ‘file’, ‘link’, ‘timestamp’, and ‘linktype’ element.

```
if($result){
		$num = mysql_num_rows($result);
		$array = "";
		for($i=0;$i<$num;$i++)
		{
			$row = mysql_fetch_array($result);
			$array[$i]['id'] = $row['id'];
			$array[$i]['start_time'] = $row['start_time'];
			$array[$i]['end_time'] = $row['end_time'];
			$array[$i]['name'] = $row['name'];
			$array[$i]['file'] = $row['file'];
			$array[$i]['link'] = $row['link'];
			$array[$i]['timestamp'] = $row['timestamp'];
			$array[$i]['link_type'] = $row['link_type'];

		}
		return $array;
	}
```

The web service is now ready to retrieve all the information necessary for the client. To manually invoke the web service we can write the address in a browser’s address bar as: http://130.237.15.244/webservices/web__service.php. This results in the output shown in **Figure 8-6**. This screen shot shows the name of our web service “Digital Display Information System”, a link to WSDL operations, and another link to the function “information”.

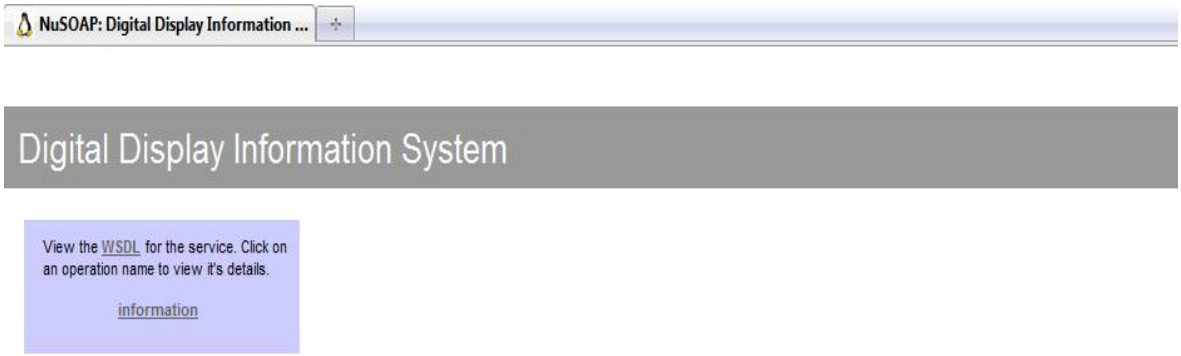

Figure 8-6: Invoking the web service

By clicking on the “information” link on the above screen SOAP request will be made by the client and the server will return the information, which will be shown by the browser. This will show all the details of the invocation of the function “information” as shown in 8-7.

Alternatively, by clicking on the “WSDL” link, an XML file will be displayed with a description of the web service and all other important operations such as namespace, port type, SOAP operation, SOAP action, and operation name information.

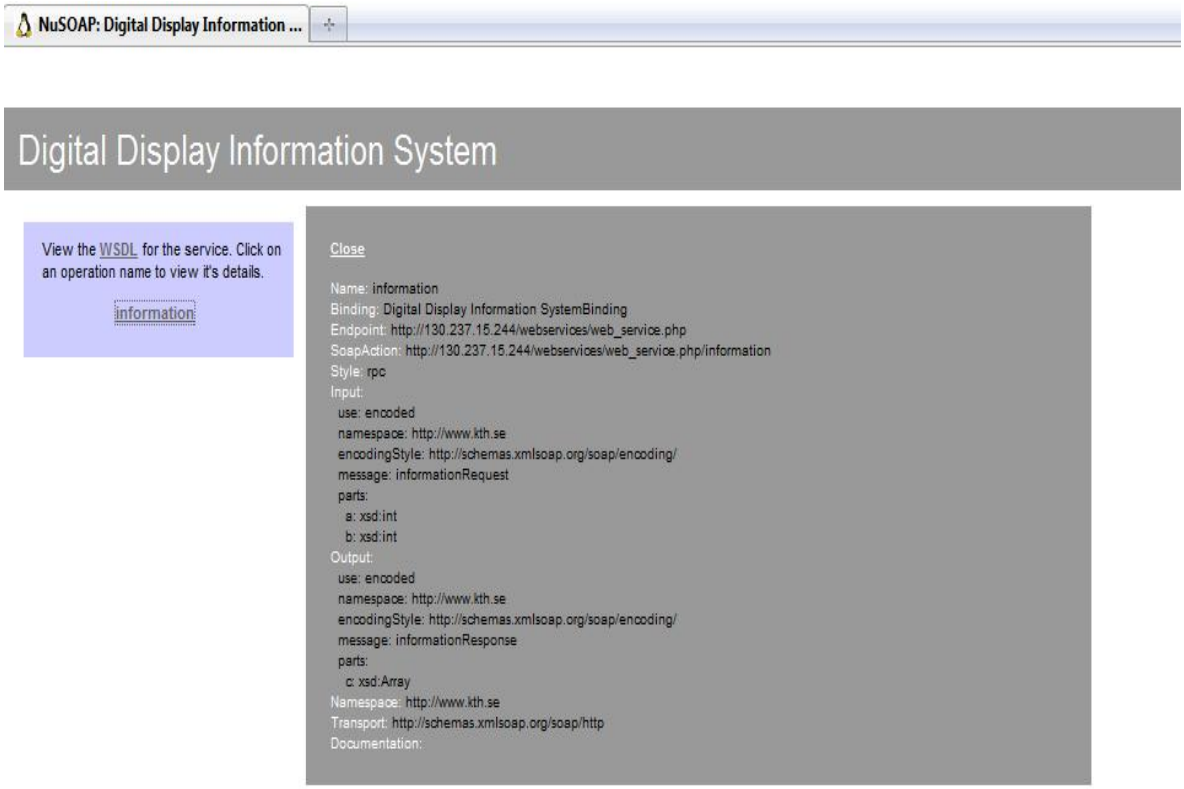

Figure 8-7: Result of invoking the web service function "information" details

8.2.2 Creating a SOAP Client

Now that we have a working web service, we need to create a SOAP client in order to automatically communicate with the SOAP server. The SOAP client execute on the client machine, i.e., the machine attached to the display. First we load the SOAP libraries that implement the functions we are making available via the web service, then we will create a SOAP client that will send a request to the SOAP server.

In PHP we can do this by creating a new instance of the `nusoap_client` based on the WSDL for our web service. The resulting SOAP client will be stored in the variable `$SOAP`. The first parameter to the `nusoap_client` function is the URL of the WDSL description of the web service. The second parameter is set to `true` indicating that a WSDL description exists. In the first parameter we give the IP address of our server, because the host name is not in the local DNS table. When the SOAP client calls the web service, the web service will fetch the desired information from the Digital Display Information System's database and send it in the response to the SOAP client. Figure 8-8 clearly shows how SOAP client works.

```
require_once 'lib/nusoap.php';

$soap = new nusoap_client('http://130.237.15.44/webservices/web_service.php?wsdl', true);
```

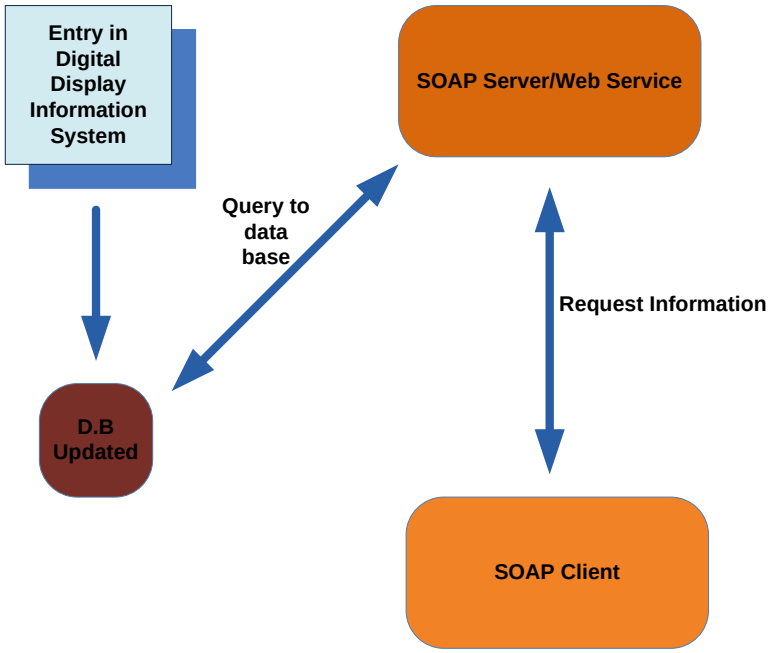

Figure 8-8: The operations performed by the SOAP Client

Every display has a unique ID, and the ID of the display is stored in the database in the table mrbs_display. On the basis of that ID all the information will be retrieved from the database through the web service. We can also send a request to web service based on the static IP or MAC address of a specific display to retrieve the contents. But there will be a problem if any change came into the IP of a particular display it will require a major change to our system. MAC addresses are difficult to manage and maintain because it is difficult for administrator to remember the MAC address of a specific display, e.g the information like which particular MAC address belongs to which display and where the location of that specific display is. So we use a simple approach by assigning a unique ID (number) to each display because we are using the (MRBS software) where for each and every room they used the simple unique ID. Thus on each client we have created a XML file in which we indicate the ID of the specific display attached to this client (see Figure 8-9). When the SOAP client calls the web service it passes the unique ID of the display as contained in this file.

```
<data>
<record>
<id>1</id>
</record>
</data>
```

Figure 8-9: The contents of the file "display_ID.xml"

If we want to show some URLs or files on the display which is located at the entrance to Wireless@KTH, then we have to give the ID of that display (which happens to have the value 1). If we wanted to show something on the display which is located at TSLAB in the Forum building, then we specify the ID value as 10. The list of displays and their IDs is stored in the database table mrbs_display.

The schedules returned by the web service will be stored in another XML file (named "data.xml"). An example of the information that might be stored in this file is shown in Figure 8-10. The next section will describe how the information in this file is used to specify the content to be displayed. Figure 8-11 shows an overview of the system where on the client side (PC attached with display) SOAP client runs on web server and this web server is also used to cache the contents. XML file where all the requested schedules are stored. On the server side web service/SOAP server is running. The following steps will illustrate we have described thus far. These steps are:

1. SOAP client request via HTTP to web service.
2. Web service retrieve information from data base.
3. Web service response to SOAP client.
4. SOAP client saved all information in XML file.

```
<information>
<record id="178">
<start_time>1254983400</start_time>
<end_time>1255015800</end_time>
<name>www.tslab.ssvl.kth.se/</name>
<link>www.tslab.ssvl.kth.se/</link>
<file/>
<link_type>1</link_type>
<timestamp>2009-10-08 10:50:29</timestamp>
</record>
<record id="177">
<start_time>1254985200</start_time>
<end_time>1255014000</end_time>
<name>KTH Logo</name>
<link/>
<file>kth_logo.gif</file>
<link_type>2</link_type>
<timestamp>2009-10-08 10:48:54</timestamp>
</record>
```

Figure 8-10: Information received from server will be stored in the file "data.xml"

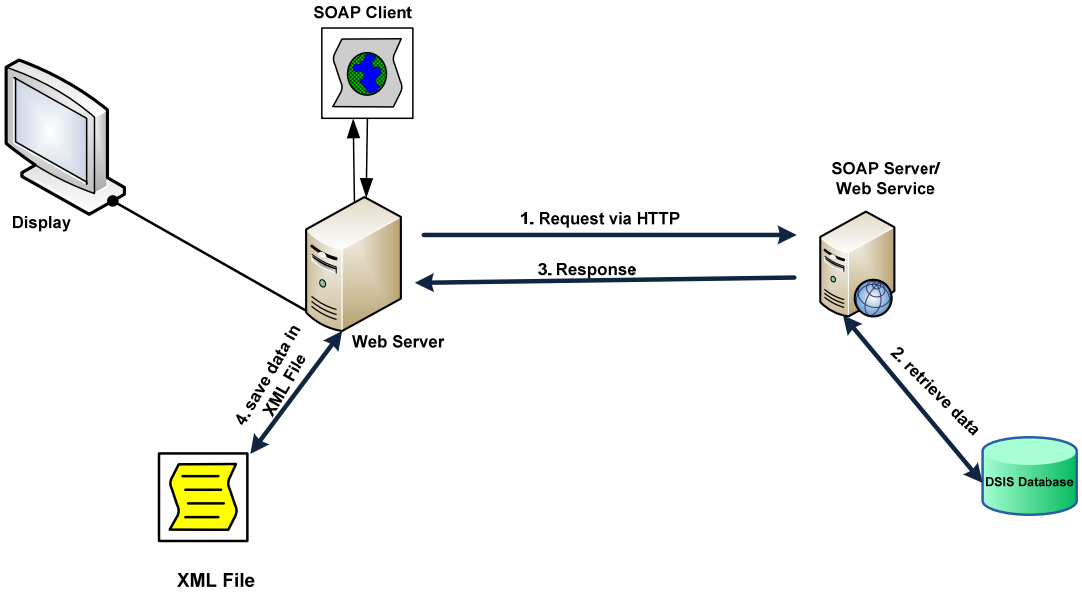

Figure 8-11: System Overview

8.3 Displaying Contents

Now that we have the information to be displayed (i.e., the contents), we need some software on the client machine to display the contents according to the specified schedule. This software keeps track of some important information for administrators, such as, exactly what is being shown at each digital display (i.e., "the current activities"), coming activities, and scheduling information about these activities.

This schedule has four important fields: URL/File, start time of the content, end time of the content date, and date/time of when this specific content was entered into the main digital display information system. Figure 8-12 shows an example of this as displayed by the client side of the system. Note the four fields that have been specified.

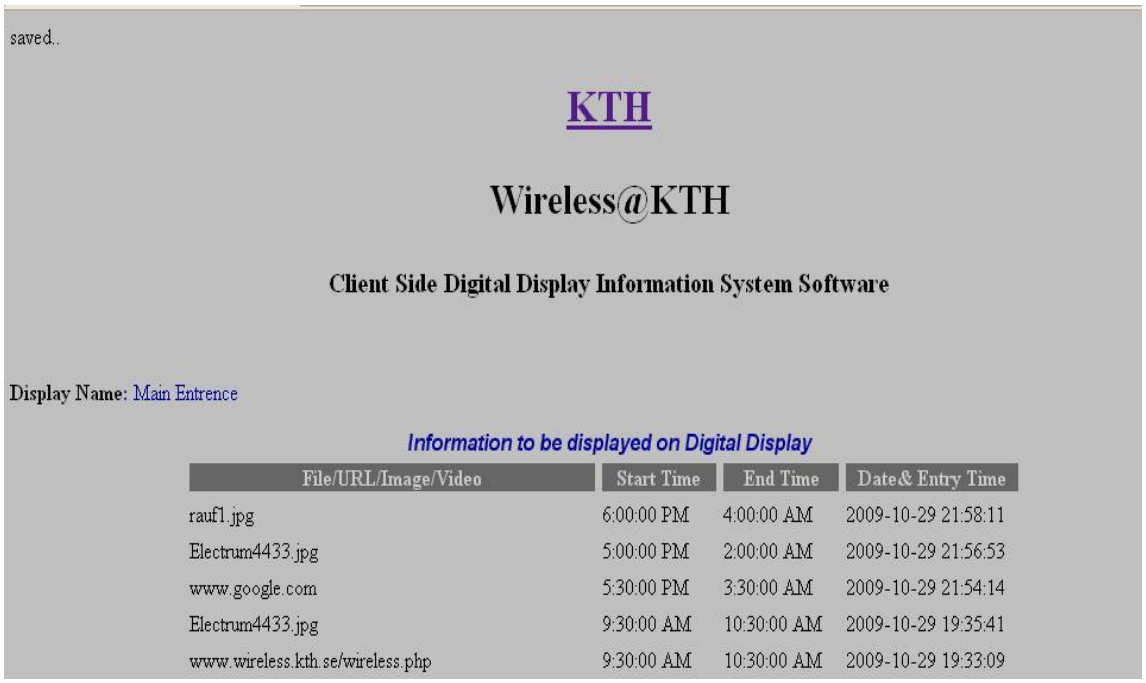

Figure 8-12: Client Side Digital Display Information System Software

This client software extracts the necessary data from the XML file ("data.xml"). We have placed a meta function on the web page of this software and also placed the source code of our SOAP client there. As shown below, after 10 seconds the meta function will be invoked to refresh this page. When the browser refreshes this page the SOAP client will be invoked. This client will call the web service, which will send the new schedules that is stored in the XML file. These schedules will be used to determine what contents will be displayed on the digital TV and when it will be displayed.

```
<meta http-equiv="refresh" content="10" >
```

In order to display the contents we have created another web application with the name "display.php". This application uses the data from the XML file and compares the current (local) time with the scheduled time for each item to be displayed. This code is shown below:

```
if(time() >= $start_time && time() <= $end_time)

echo "\n<meta http-equiv=\"refresh\" content=\"5;http://localhost/webservices/display.php?count=".$count."\">\n";
```

Note that the code also generates a new meta function to cause the display.php file to be called every 5 seconds. The refresh rate of 5 seconds and 10 seconds are arbitrary chosen and the two web applications (clientview.php and display.php) will not cause any read/write conflicts on data.xml file. The current system is designed to support only web contents, static images, and text; this allows the refresh rate to be statically specified. Although our system can be extended to video/flash contents and the refresh rate can be dynamically updated according to the size of the media file. If the type of the link is 1, then the specified URL will be displayed, otherwise the contents of the file will be displayed. See the following source code fragment.

```
if($link_type[$i] == 1)
                        {
                                $links = $record->getElementsByTagName( "link" );
                                $link[$i] = $links->item(0)->nodeValue;
                        }
                        else
                        {
                                $files = $record->getElementsByTagName( "file" );
                                $link[$i] = $files->item(0)->nodeValue;
                        }
                        $i = $i + 1;
                }
        }
```

Security benefits and advantages such as low complexity and being easy to implement caused me to employ a polling base approach in our prototype digital signage system. An event based approach has more security vulnerabilities than a polling based approach (as the polling approach can exploit the existing mechanisms for providing security via TLS to HTTP). The complexity of an event based approach is high. Additionally, the polling based approach allows the server to know that each client is functioning (or at least that it is continuing to poll for updates). Therefore, the system uses a polling based approach instead of an event based approach, with every client retrieving its schedule from the server after 10 seconds.

We used the concept of frames to display multiple (different) web pages in the same browser window. This is very important as we want to display different URLs within the same browser window. Note that we cannot simply tell the browser to display a particular URL or we would lose control of the browser (i.e., it would only display the specific page). Additionally, we have designed this system to support all kinds of web browsers, hence we could not use a browser specific means to control the browser. Therefore, in order to display all the specified pages in a browser automatically we simply have the browser display a single URL: http://localhost/webservices/display.php. Here localhost is the name of our client machine (i.e., the browser will use the loop back IP address for the IP address of the web server), webservices is the directory name -- this will actually be the directory ApacheTriad/htdocs/webservices, and display.php is our web page which will dynamically update and display the scheduled information, i.e. display.php is our web application page which will be running in the browser all the time. We will set the meta information for this web page so that every time it runs it will display a new URL for a period of 5 seconds. We have a counter variable so that this web application will display the next URLs in the defined sequence. By default the first URL in every sequence will always be the main web page of KTH (i.e., www.kth.se).

```
$i = 0;
  $link_type[$i] = 1;
  $link[$i] = "www.kth.se";
  $i = $i + 1;
```

It is important to note that client's web server locally stores and fetch the files to be displayed from the directory: mrbs/web/files. In order to display these files, we must specify the full path to these files. Note that in our current implementation these files are stored on the main server machine (where our main Digital Display Information System is running). Therefore, we must give the full path including the IP address of our server machine. An example is shown below:

```
<img src="http://130.237.15.244/mrbs/web/files/<?= $link[$random]?>" >
```

Note that the client's web server will automatically cache these files and will also automatically replace them with a new version if the file on the server machine has a later modification time than the copy that is in the local cache. This provides a lot of nice functionality that we did not have to implement and it also means that if the server machine fails, we will be able to continue to display content – but this content might be stale.

As a result of the caching above, if the main server fails, then there will be no immediate effect on our system because we have all the information (schedules) stored in a local XML file. The contents will be shown on the digital display until the end of schedule. When there are no longer scheduled contents to be displayed, then the display will simply show the main page of the KTH website (as mentioned above). In the scheduled contents are not cached then if this contents can not be retrieved it can not be displayed, but the previous contents will continue to be displayed until the 5 seconds is up, then the next scheduled item will be displayed.

8.4 Deployment

In order to deploy and test our system we first deployed it in the laboratory for testing, then we deployed it by attaching the client machine to the large display located at the Wireless@KTH center in the Electrum building in Kista.

This required the following steps and procedures:

1. We began by deploying the server software on a server machine (with the IP 130.237.15.244). This was simply done by copying all of our files and folders (for the Digital Display Information System and webservice) to that machine. Placing the files in the folder /src/www/htdocs as shown in Figure 8-13.

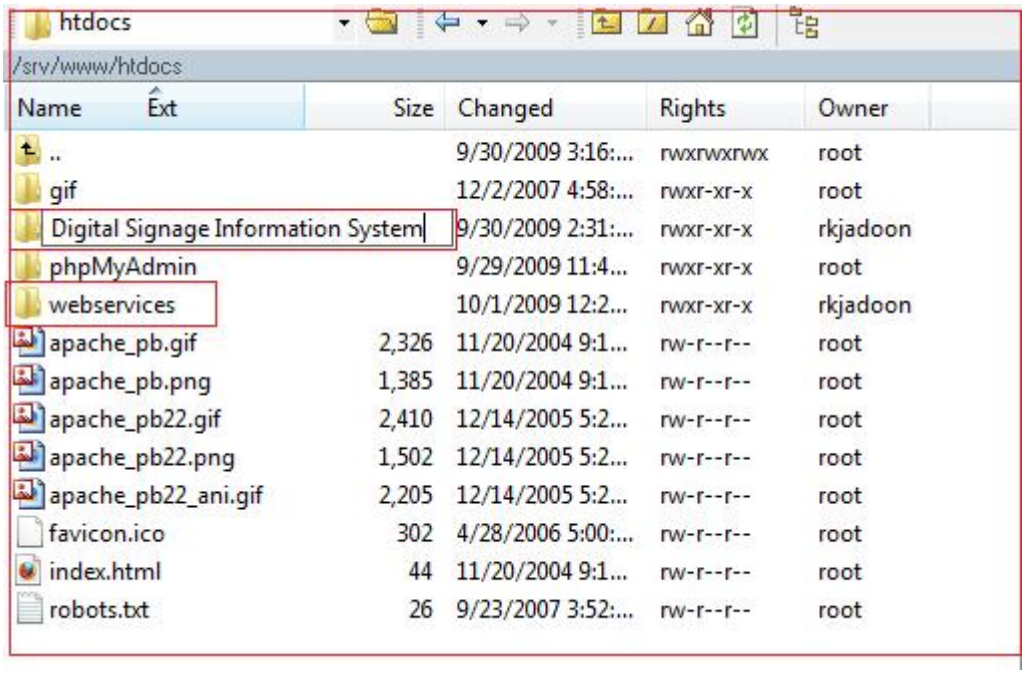

Figure 8-13: Main Server Directory Files and Folders

2. Next we used PHPMyAdmin to create our database and tables. Following this we set the username and password needed by our software to connect to the database. This is done by

setting the following lines in the file "Config.inc.php": **srv/www/htdocs/mrbs/web/config.inc.php**

```
// Database login user name:
$db_login = "root";
// Database login password:
$db_password = 'xxxxxxxxxxxxxxxxxxxx';
```

3. Next we copied all of our web service files into the folder webservices (this included the folder Lib, Samples, displayid.xml, and web_service.php). In the web_service .php file it is very important to change the parameters for database connectivity to specify the username **root** and the same password as configured above.
4. In the next step we installed all the folders and files on the client machine (this includes the file client.php, data.xml, clientview.php, and the webservices files Samples and Lib). Note that before installing these files we locally installed an Apache web server to run our application.
5. The next step is to access and run these applications on the server machine. We do this by specifying the complete path: http://130.237.15.244/mrbs/web . The main page of the software will be displayed and you need to enter a valid username and password. Note that we can create as many usernames and passwords as we need by placing them in the Config.inc.php file of this software. After giving a valid username and password you can now enter information about the content to be displayed, where and when to display it, etc.
6. The final step is to display the specified content on the display. To do this we need to run two of our web applications on the client attached to the display. The first web page to visit in a browser is "clientview.php" – so that our SOAP client will call the web service located on the server machine and store the relevant information about what is to be displayed in the local data.xml file. To do this you simply enter the following URL into the browser address window: http://localhost/webservices/clientview.php. Next we invoke the web page http://localhost/webservices/display.php to start the process of continuously displaying the scheduled content.

Chapter 9 - Evaluation

In this chapter we evaluate our work in terms of performance and reliability. In addition we will compare it with the existing digital signage software being used by the Wireless@KTH center[**]. We will also discuss whether we achieved our goals and remark upon the methods used during the entire project.

9.1 Were the goals achieved?

At the beginning of this research project two goals were presented regarding the development of a digital signage system. The first goal was to design, developed, and evaluates a hardware/software solution based on a PC, to control and display different web contents (or other dynamic information) on digital displays located at different locations. We have achieved our first goal by successfully implementing a digital display management system through which a user can show any dynamic information on a digital display at Wireless@KTH

Our main focus was to design a system which is easy to use. We wanted to have a friendly interface, so that users would not face a steep learning curve in order to use the system. However, the most important requirement was that the system should provide timely and relevant information, in the form of content disseminated by the digital signage system.

Based upon the fact that the software has basically the same look and feel, the user interaction with this system is very similar to a web based information system that the staff is already using for room booking. Thus we think that we have met the goals of the system being easy to use and having a friendly user interface. The result as we noted earlier is that there is almost no additional training required to use the system.

With regard to the timeliness of the information, the content is displayed in the next cycle of pages following when the content is entered and scheduled at the server. This is as timely as any system could display the information. It should be noted that in comparison with the existing commercial system – it is easier to generate dynamic content (for example based upon the current room booking system) with the system developed in this thesis project.

The "Digital Display Information System" performs the desired functions. Specifically it provides:

- Content Management.
- Content Scheduling
- Display Management.

The second major goal of this thesis project was designed a system such that if the server crashes or fails, the rest of the system will continue to operate, i.e., that the clients can display the existing information that they have. We have met this goal due to our distributed design. Additionally, the use of a local Apache server running at the client provides automatic cache management for the content – thus there is no need to explicitly transfer content to the client systems. Thus the system can even be used with a browser that does not do caching itself, while still getting good caching performance. Note that because the content is being retrieved by the browser from the local Apache server, there is no

[**] Note that this digital signage software was bought and installed independently of this thesis project.

network latency for this communication. The content is stored on the server until the first that it is to be displayed on the digital signage. Downloading this content generally takes very little time, i.e less than 5 sec, as the network latency is low for this communication between server and digital signage. However, a side effect of having a local Apache server is that if the browser is also doing caching, then there will be two copies of the contents stored on the local machine.

In addition to these two goals we also added a mechanism to turn off the digital TV display automatically at the end of the day's schedule. We are able to control the digital TV via its serial port, thus we can turn the display off automatically. Unfortunately, the same interface is **not** able to turn the display on automatically at the start of the day – thus the display still has to be manually turned on.

9.2 Architecture of Digital Signage System

Figure 9-1 shows the architecture of our digital signage system (as both designed and implemented). The system is a distributed client server architecture where the client sends requests to the web service, the web service requests data from the scheduling database and sends back to client, the client format and displays this content using a local browser. Different clients requests to the server containing the id of their display, and receive the content that this client is scheduled to display. All clients communicate directly with server and do not need to communicate with each other. Thus the clients are independent and not interdependent, so the failure of any client will not affect the other clients or the server.

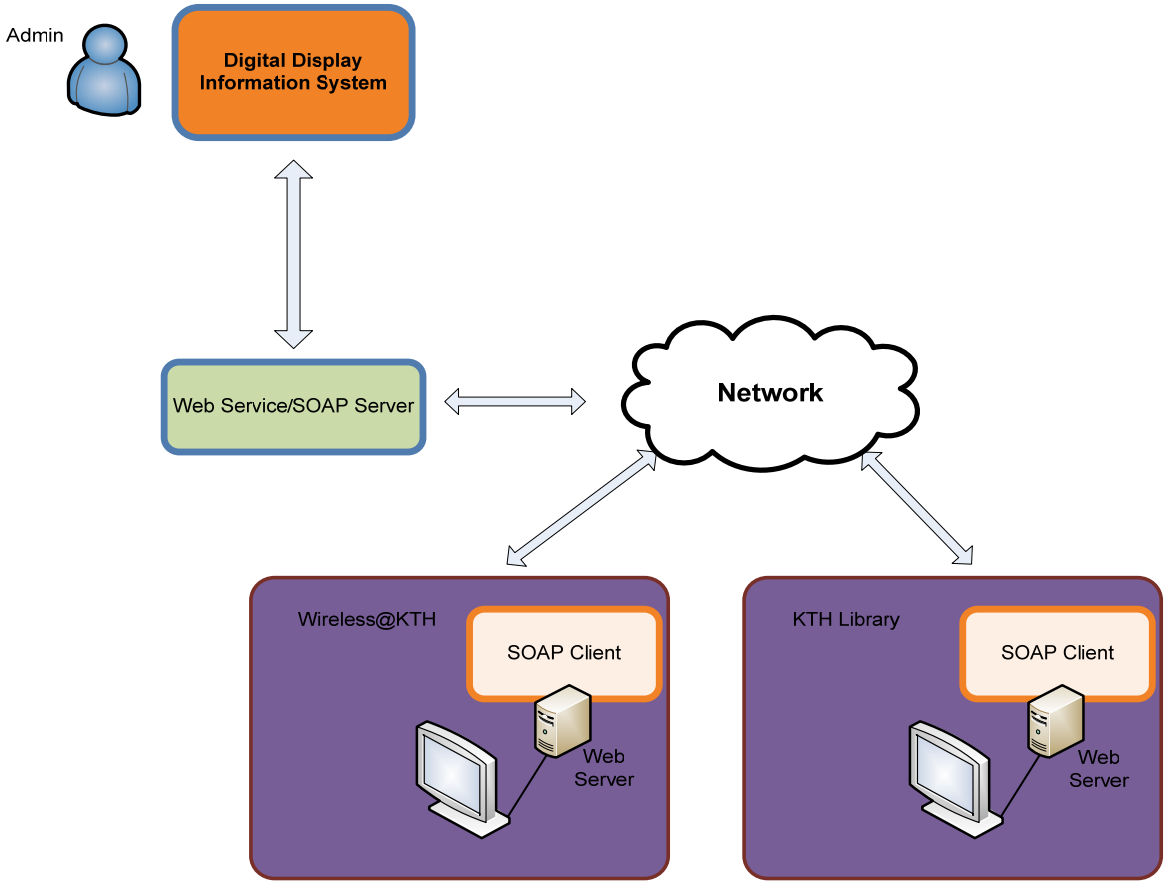

Figure 9-1: Architecture of the Digital Signage System

The proposed solution is not confined to any particular type of organization; rather it could be deployed by any organization – ranging from a setting with a single display to a distributed organization with thousands of displays. In this digital signage system, the contents of the display are controlled directly by a local web browser. As we are using open source software as the basis for our digital signage system, anyone can mold it to their needs.

All the contents on the displays can be easily controlled by users/administrators from anywhere on the network that is able to reach this device (either directly or through a proxy). However, this has not been evaluated or studied in any detail. It represents an area where future work is needed.

Two major advantages of this system architecture are:

- If the server or its database crash, the local displays continue to display their content as scheduled. By default if there is no content scheduled it will always display a statically specified URL (in our case: www.kth.se).
- The system is very fast, with regard to the time from when the user enters content in the digital signage management system and schedules this content for dissemination on a display. The time required for a client to fetch the information is short (i.e., a matter of seconds, typically less than 5 seconds). If the content has been updated or newly created, then in the next display cycle the display will retrieve the updated or new content and display it according to the specified schedule.

Weakness of the current system:

- The security of the current system is weak. For example, there is no check that a client request actually comes from a valid client or that this client is requesting the data for its own display and not that of another display. (This suggests the system and the clients should have SSL certificates and use TLS or the systems should be interconnected via VPNs.)
- There has not been a performance measurement done of the actual delay between content being first placed on the server or updated there, and the time until the client displays the updated information.
- The 5 or 10 second timer values were chosen arbitrarily and it is not clear that these are the most suitable values.
- A secure configuration of the main server and the local Apache servers have not been examined or specified.
- No measurement of the system reliability was actually carried out. While the server machine is highly redundant and has been running continuously for 258 days, this server and the network infrastructure are on regular power mains and not attached to an uninterruptable power supply (with a backup generator); however, if the power were to go out there would be no power for the PC attached to the display or even for the display itself.
- There have been no measurements of the scaling of this system, for example, how many clients can a given server support, what is the maximum update rate of the system, etc.
- There is no mechanism specified on the client machines i.e. to inform the administrators about the server failure notifications.

9.3 Comparison with the existing systems

Currently Wireless@KTH is using software called **Smart Sign** [45] to display content on a specified schedule on a digital display. This application software has been used by many organizations. We will compare this system with our proposed solution from several

aspects. Table 9-1 shows a comparison of our system with Smart Sign. This comparison shows that our proposed solution is scalable, easy to manage, efficient and most importantly (for many organizations) is more cost effective than the Smart sign software.

Table 9-1: Comparison of our Solution with Smart Sign

		Digital Display Information System	Smart Sign
Efficiency		**Very efficient to use, does not require any configurations or settings before entering content.**	**Efficient, but require some settings such as Server name etc.**
Usability		**Easy to use with a simple and friendly web based interface. Any user or administrator can enter and manage the content and schedule from anywhere at any time.**	**Easy to use interface, but cannot support web technology. Users can only access a specified range of smart sign server machines.**
Media Support		**URLs, Images (JPEG, GIF, PNG, TIFF, BMP), Adobe PDF, … . In fact, anything that the browser can display.**	**Microsoft Power Point (PPT, PPS), URls Images (JPEG, PNG, TIFF, BMP), Adobe PDF, Video (MPEG, AVI, WMV).**
Management Expertise		**Need very little management expertise from the user point of view.**	**Needs little management expertise, but if there is any problem (software or hardware) that cannot be managed by local system administrator, the only recourse is to contact the software vendor.**
Browser Access		**Browser based solution; can support all types of browsers.**	**Does not support any browser.**
Ease of Development	**Available Software**	**Digital display information system, Apache, MySQL, … -- all open source software.**	**Proprietary vendor software**
	Operating System	**Can support and run on nearly any operating system (Linux, Microsoft's Windows XP and Vista, various UNIX systems, etc.).**	**Do not support Linux operating systems, can only run on Microsoft's Windows XP or Vista operating systems.**
	License	**Does not require any license**	**Licensed software.**
Cost	**Software Cost**	**No software cost.**	**License fee**
	Hardware Cost	**Minimal systems requires one PC (with two virtual clients), or two PCs; one as a server and the other for client and display interface. No special hardware is required.**	**Need PCs or a special hardware Media Player.**

9.4 Summary of Evaluation

There are many benefits of a web based approach to digital signage. These benefits include efficiency, scalability, low system cost, and low complexity. The best feature is that it can support all kinds of web browsers. The web browser is used to present the contents, thus any type of contents that the selected web browser (possibly with additional plug-ins) supports can be displayed.

In comparison with the two other approaches (Stand-alone and IPTV based), the web based approach is scalable in that the rendering of the contents is done locally - allowing each sign to display information specific to this display. The system is also scalable in the sense that we can easily add more displays to the system. As already said, unlike the case for IPTV approach, the failure of any display has no affect the other client or server. The system can display up to date information according to a specified schedule unlike the Stand-alone approach which can only show the contents that were loaded, hence the contents are fixed until someone physically comes to the computer and loads new contents. A digital signage system based upon a web based approach can be easily deployed and new contents and schedule can be provided by remote access from any web browser. Unlike the IPTV based approach which requires a large and complex network infrastructure and streaming IPTV media server, the proposed system requires more limited network connectivity (and potentially generates much less traffic) and simply needs a web server rather than a media server. The digital signage system I have designed is easy to use and has a simple user friendly interface through which any user or administrator can easily manage the contents and schedule from anywhere at any time. Due to these benefits, I strongly believe that such a web based approach to digital signage can replace the existing signage system in the market.

Chapter 10 - Conclusions and Future Work

10.1 Conclusions

In this thesis project I have successfully implemented a complete digital display information system. I have designed, developed, and evaluated a hardware & software solution based on a PC, to control and display different web contents (or other dynamic information) on digital displays located at different locations. The system is based upon a set of open source software packages and a few modifications to one of the packages. The system meets our original design goals. On the surface the system appears to be an efficient and scalable way to display dynamic contents via digital displays according to a specified schedule.

10.2 Future Work

There are quite a number of issues concerning the security of the system that need to be examined and addressed. There is also a need to do detailed performance measurements. The claims of efficiency and scalability are weak and actually need to be evaluated. A possible area for improvement is to pre-fetch content that is *unlikely* to be modified.

In addition there are a number of areas for future work that would make the system more attractive and competitive with commercial digital signage systems. Three of these are discussed in the following subsections.

10.2.1 Zoning

To display multiple types of information simultaneously on a digital signage, the screen can be divided into multiple regions or zones. Each zone or region can support multiple elements of contents and these different elements can even use different formats. Each and every zone or region is an independent component that might be updated or changed. In this case, web contents, video streaming of news, graphical animation giving current weather reports, and some textual information (stock prices and volumes) about different companies, and the time of day might all be displayed simultaneously.

Supporting zones would be of value to sometime organizations as they want to display multiple elements of information on a single screen rather than having the information spread out over several screens (due to the expense) or spreading the information out over time – by having it displayed on different screens in an information cycle.

One potential means of doing this is using a windowing system that allows multiple applications to share a display. Another potential means of doing this is having the separate applications render to different off screen buffers, then block copying the relevant information from these different applications to zones on the screen.

10.2.2 Media Support

In this thesis project our design only included support for three types of media (web contents (URLs), images (JPEG, GIF, PNG, TIFF, BMP) and Adobe PDF). This implementation does not have support for video, Microsoft power point slides, Excel spreadsheets, or Word files . Further work is needed to support these other media.

Therefore, support should be added for video (MPEG, AVI, WMV), Microsoft Power Point (PPT, PPS, PPTX), Microsoft Excel (XLS), and Adobe flash (SWF) contents.

Note that there are plug-ins that can display flash in a browser window, this might already be usable; but this has not been investigated. If zoning is supported such that different applications can be used to generate visual output, then most of these other media could be supported (rather easily).

10.2.3 Backup Server

There should be an easy means of setting up a backup server and offering hot-failover to this alternate server – should the main server fail (for example, using a network dispatcher – as is often used to support load balancing across a set of web server). While there are existing solutions to exploit a distributed database, the main difficulty would be how to change the protocol that the clients use to locate the currently active server. Possibilities include configuring multiple servers into the client software or using a protocol such as SCTP to communicate with what would look like a multi-homed server, but could actually be distributed replicates of the main server.

References

[1] Cisco Digital Media System: Comprehensive. Scalable. Network-Centric. A white paper by Cisco,2008.[WWW] http://www.cisco.com/en/US/prod/collateral/video/ps9339/ps7063/ps7066/prod_brochure0900aecd80537ce5.pdf

[2] Wikipedia. [WWW] Last access 2009-04-29 http://en.wikipedia.org/wiki/Digital_Signage.

[3] Scala Inc. [WWW] Last access 2009-04-29 http://www.beaver-group.com/files/pdf/digital-signage.pdf.

[4] Cisco. [WWW] Last access 2009-04-30 http://www.cisco.com/en/US/products/ps7220/index.html.

[5] Daily DooH. [WWW] Last access 2009-05-06 http://www.dailydooh.com/archives/8029

[6] Wikipedia. [WWW] Last access 2009-05-07 http://en.wikipedia.org/wiki/MPEG.

[7] Wikipedia. [WWW] Last access 2009-05-07 http://en.wikipedia.org/wiki/Adobe_Flash

[8] Vital CAST, Dynamic digital signage that commands attention, A white paper, 2008.

[9] Cisco, Digital signage distribution methodologies overview, A white paper, 2007 [WWW].http://www.cisco.com/en/US/prod/collateral/video/ps9339/ps7220/prod_white_paper0900aecd805acf12.html

[10] Cisco, Cisco Digital Media Players,[WWW]. Last access 2009-05-08 http://www.cisco.com/en/US/products/ps7220/prod_models_home.html

[11] Cisco, Cisco Digital Media Player, 4400G, [WWW]. Last access 2009-06-04 http://www.cisco.com/en/US/products/ps9639/index.html.

[12] Cisco, Cisco Digital Media Player, 4305G, [WWW] Last access 2009-06-04. http://www.cisco.com/en/US/products/ps8767/index.html.

[13] Cisco, Cisco Digital Media Manager, [WWW]. Last access 2009-05-11 http://www.cisco.com/en/US/products/ps6681/index.html.

[14] Stinova Ltd, [WWW]. Last access 2009-05-11 http://www.stinova.com/index.php/about.

[15] Stinova Ltd, digital signage soloutions, integrated IPTV technologies, A white Paper, 2008. [WWW]. WWW.Digital-Signage.cc.

[16] Media Star Evolution, [WWW]. Last access 2009-05-12 http://www.cabletime.com/#

[17] CAYIN Technology,[WWW]. Last access 2009-05-12 http://www.cayintech.com/

[18] CAYIN Technology,[WWW]. Last access 2009-05-25 http://www.cayintech.com/digital_signage_player_smpwebduo3.htm.

[19] AVT (Digital Media System), Digital Signage ROI, A white paper. July, 2008.

[20] Ben Phillips, Mac Joye, Sabrina Gobin, Daniel Arcila and Matt Nozemack. Digital Signage the digital value preposition, S.O.B Incorporated.

[21] L.Terry, Feast for the eyes, business: Restaurateurs begin embracing digital signage, Restaurant News, www.nrn.com. Last access 2009-05-25.

[22] Rui José, Beyound Application-Led Research in Pervasive Display Systems, Presentation at the Workshop on pervasive display infrastructures, interfaces and applications, Dublin, Ireland, May 7th, 2006 http://ubicomp.algoritmi.uminho.pt/perdisplay/docs/Jose-Perdisplay-jose.pdf

[23] S. Ljungblad, Informative Art. in Workshop on Collaboration with Interactive Walls and Tables. 2002.

[24] J. Stasko, et al. Personalized Peripheral Information Awareness through Information Art. in Sixth International Conference on Ubiquitous Computing (UbiComp 2004). 2004. Nottingham, England: Springer.

[25] N. Streitz, et al., Ambient Displays and Mobile Devices for the Creation of Social Architectural Spaces, in Public and Situated Displays: Social and Interactional Aspects of Shared Display Technologies, K. O'Hara, Editor. 2003, Kluwer. p. 38-409.

[26] C. Wisneski, et al. Ambient Displays: Turning Architectural Space into an Interface between People and Digital Information. In proceedings of the First International Workshop on Cooperative Buildings (CoBuild '98). 1998. Darmstadt, Germany: Springer.

[27] S. Izadi, et al. Dynamo: A public interactive surface supporting the cooperative sharing and exchange of media. UIST 2003. November 2003.

[28] D.M. Russell and A. Sue. Using Large Public Interactive Displays for Collaboration. in Workshop on Collaboration with Interactive Walls and Tables. 2002.

[29] John V. Harrison and Anna Andrusiewicz. Enhancing Digital Advertising Using Dynamically Configurable Multimedia. International Conference on Multimedia and Expo. IEEE, Volume 1, July 2003, pages 717-720.

[30] John V. Harrison and Anna Andrusiewicz. Using Wireless Networks to Enhance Narrow Casting in Public Places. First IEEE Consumer Communication and Networking Conference CCNC 2004, January 2004, pages 600-604.

[31] John V. Harrison and Anna Andrusiewicz. An Emerging Marketplace for Digital Advertising Based on Amalgamated Digital Signage Networks. IEEE International Conference on E-Commerce Technology. June, 2003, pages 149-156.

[32] Jörg Müller, Juliane Exeler, Markus Buzeck, and Antonio Krüger. ReflectiveSigns: Digital Signs That Adapt to Audience Attention. Proceedings of 7th International Conference on Pervasive Computing, Springer-Verlag Berlin Heidelberg LNCS 5538, May 2009, pages 17-24.

[33] C. Küblbeck and A. Ernst, Face detection and tracking in video sequences using the modified census transformation. Image and Vision Computing 24 (6), 564–572, 2006.

[34] Jörg Müller, Dennis Wilmsmann, Juliane Exeler, Markus Buzeck, Albrecht Schmidt, Tim Jay, and Antonio Krüger. Display Blindness: The Effect of Expectation Attention towards Digital Signage. Proceedings of 7th International Conference on Pervasive Computing, Springer-Verlag Berlin Heidelberg LNC 5538, May 2009, pages. 1–8.

[35] D. Jankowicz,: The Easy Guide to Repertory Grids, Wiley, Chichester. 2003.

[36] Jörg Müller, Alex Schlottmann, and Antonio Krüger. Self-optimizing Digital Signage Advertising, Adjunct Proceedings of Ubicomp 2007, September 2007 http://ifgi.uni-muenster.de/~muellerj/publications/SelfOptimizingDigitalSignage.pdf

[37] MRBS: Introduction and downloaded. last access at 25/06/2009 http://mrbs.sourceforge.net/

[38] Apache: Download Mirrors: Last access at 25/06/2009 http://www.apache.org/dyn/closer.cgi

[39] PHP: downloads. Last access at 25/06/2009 http://www.php.net/downloads.php

[40] MySQL: Community Downloads. Last access 25/06/2009 http://dev.mysql.com/downloads/

[WWW]. Last access 27/06/2009

[41] http://www.phpmyadmin.net/home_page/index.php.

[WWW].last access 28/09/2009

[42] http://www.w3schools.com/webservices/ws_intro.asp

[WWW]. Last access 28/09/2009

[43] http://en.wikipedia.org/wiki/SOAP#Request

[WWW]. last access 28/09/2009

http://www.w3schools.com/WSDL/default.asp

[44] [WWW]. last access 20/08/2009 http://sourceforge.net/projects/nusoap/#

[45] [WWW]. last access 19/10/2009

[46] http://www.smartsign.se/display.aspx

[WWW]. Last access

A. Module Changes in MRBS Software

In this and the following appendices we will only show that code which we have added or modified in the MRBS software.

 # editentry.php(Following is the code we have added to editentry.php, which select URL and upload files)

```
<?php
// $Id: edit_entry.php 1002 2009-01-23 20:36:13Z cimorrison $
require_once('grab_globals.inc.php');
include "config.inc.php";
include "functions.inc";
include "dbsys.inc";
include "mrbs_auth.inc";

global $twentyfourhour_format;
// Get form variables
$type_rkj = 0;
if (isset($id))
{
  $sql = "select name, create_by, description, start_time, end_time,
         type, room_id, entry_type,link,file,link_type, repeat_id from
         $tbl_entry where id=$id";

  $res = sql_query($sql);
  if (! $res)
  {
    fatal_error(1, sql_error());
  }
  if (sql_count($res) != 1)  {
    fatal_error(1, get_vocab("entryid") . $id . get_vocab("not_found"));
  }


  $row = sql_row_keyed($res, 0);
  sql_free($res);
  $type        = $row['type'];
  $room_id     = $row['room_id'];
  $entry_type  = $row['entry_type'];
  $rep_id      = $row['repeat_id'];
  $type_rkj = $row['link_type'];
  $file_rkj = $row['file'];
  $link_rkj = $row['link'];
  <form class="form_general" id="main" action="edit_entry_handler.php"
   method="post" enctype="multipart/form-data">
  <fieldset>
  <legend><?php echo get_vocab($token); ?></legend>

    <div id="div_name">
```

```
      <label for="name"><?php echo get_vocab("namebooker")?>:</label>
      <input id="name" name="name" value="<?php echo
       htmlspecialchars($name) ?>">
       </div>

   <div id="div_description">
      <label for="description">Full Description</label>
   <!-- textarea rows and cols are overridden by CSS height and width -->

      <textarea id="description" name="description" rows="4"
cols="4"><?php echo htmlspecialchars ( $description ); ?></textarea>
       </div>
<div id="rep_type">
        <label>Type:</label>
        <div class="group">

// defined two fields with their type, if type=1 then it will be URL and
if Type=2 it will be file
 <label><input id="id1" class="radio" name="type" type="radio" value="1"
onclick="changeLink(this);" <? if($type_rkj == 1) echo
'checked="checked"';?> >
Link</label>
<label><input id="id2" class="radio" name="type" type="radio" value="2"
onclick="changeLink(this);" <? if($type_rkj == 2) echo
'checked="checked"';?>>
 File</label>
       </div>
      </div>
<div id="div_link" <? if($type_rkj == 1) echo 'style="display:block;"';
else echo 'style="display:none;"';?> >
      <label for="link">Link:</label>
   <!-- textarea rows and cols are overridden by CSS height and width -->

<input id="link" name="link" value="<?php echo
htmlspecialchars($link_rkj) ?>">
     </div>
<div id="div_file" <? if($type_rkj == 2) echo 'style="display:block;"';
else echo 'style="display:none;"';?>>
<label for="File Upload">File Upload:</label>
            <input type="hidden" name="MAX_FILE_SIZE" value="2000000">
            <input name="userfile" type="file" id="userfile">
      </div>

       <?
        if(  $type_rkj == 2 && $token == "editentry")
        {
       ?>
       <div id="div_date">
      <label><?php echo "Uploaded File";?>:</label>
            <?= $file_rkj; ?>
```

```
</div>
  <?
  }
  ?>
```

B. edit_entry_handler.php

 # edit_entry_handler.php

```
<?php
// $Id: edit_entry_handler.php 1003 2009-01-23 20:51:57Z cimorrison $

require_once "grab_globals.inc.php";
include "config.inc.php";
include "functions.inc";
include "dbsys.inc";
include "mrbs_auth.inc";
include "mrbs_sql.inc";
// Here is the Function upload
function upload_file($_FILES)
{
       $target = "files/";
       $target = $target . basename( $_FILES['userfile']['name']) ;
       //echo $target;
       if(move_uploaded_file($_FILES['userfile']['tmp_name'], $target))
              return true;
       else
              return false;
}
// Get form variables
$link_rk = get_form_var('link', 'string');
$type_rk = get_form_var('type', 'int');
$file_rk = "";
// Now to rmove the message of conflict and to enter multiple data on MRBS
software we have comment or disable the following code.
// Check for any schedule conflicts in each room we're going to try and
// book in
$err = "";
/*foreach ( $rooms as $room_id )
{
  if ($rep_type != 0 && !empty($reps))
                                        a
    if(count($reps) < $max_rep_entrys)
    {
      for ($i = 0; $i < count($reps); $i++)
      {
        // calculate diff each time and correct where events
        // cross DST
        $diff = $endtime - $starttime;
        $diff += cross_dst($reps[$i], $reps[$i] + $diff);
```

```
        $tmp = mrbsCheckFree($room_id,
                             $reps[$i],
                             $reps[$i] + $diff,
                             $ignore_id,
                             $repeat_id);

        if (!empty($tmp))
        {
          $err = $err . $tmp;
        }
      }
    }
    else
    {
      $err         .= get_vocab("too_may_entrys") . "\n";
      $hide_title  = 1;
    }
  }
  else
  {
    $err .= mrbsCheckFree($room_id, $starttime, $endtime-1, $ignore_id, 0);
  }

} // end foreach rooms*/
// If the rooms were free, go ahead an process the bookings
if (empty($err))
{
  foreach ( $rooms as $room_id )
  {
    if ($edit_type == "series")
        {
      //echo 'rauf';
         $file_res = false;
         if($type_rk == 2)
         {
              $file_rk = basename( $_FILES['userfile']['name']);
              $file_res = upload_file($_FILES);
         }

         $new_id = mrbsCreateRepeatingEntrys($starttime,
                                             $endtime,
                                             $rep_type,
                                             $rep_enddate,
                                             $rep_opt,
                                             $room_id,
                                             $create_by,
                                             $name,
                                             $type,
                                             $descripti
                                             $link_rk,
```

```
                                        $type_r
                                        $file_rk
                                        $file_res,
                                        isset($rep_num_weeks) ? $rep_num_weeks
: 0);
}
        // Mark changed entry in a series with entry_type 2:
      if ($repeat_id > 0)
      {
        $entry_type = 2;
      }
      else
      {
        $entry_type = 0;
      }

         //echo 'shujatsdfd';
$file_res = false;

         if($type_rk == 2)
         {
              $file_rk = basename( $_FILES['userfile']['name']);
              $file_res = upload_file($_FILES);
         }

      // Create the entry:
      $new_id = mrbsCreateSingleEntry($starttime,
                                      $endtime,
                                      $entry_type,
                                      $repeat_id,
                                      $room_id,
                                      $create_by,
                                      $name,
                                      $type,
                                      $description
                                      $link_rk
                                      $type_rk
                                      $file_rk
                                      $file_res);
}
```

C. Web_service.php

 # Web_service.php

```
<?php
//Load Soap libraray
  require_once 'lib/nusoap.php';
// define soap Object
  $soap = new soap_server;
  $ns = "http://www.kth.se";
// Setup and configure WSDL file
  $soap->configureWSDL('Digital Display Information System', $ns);
  $soap->wsdl->schemaTargetNamespace = $ns.'/xsd/';
//Register a function called Information
  $soap->register(
    'information',
    array('a' => 'xsd:int',
    array('b' => 'xsd:Array'),
    $ns
  );
// XML SOAP request
  $soap->service(isset($HTTP_RAW_POST_DATA) ?
    $HTTP_RAW_POST_DATA : '');
 function connect(){

return      mysql_connect("localhost","root","ccslab2");

 }

// function which performs operations on input a, and b.

function information($a) {

      $dbconn = connect();

      mysql_select_db("mrbs",$dbconn);



$result = mysql_query("Select * from mrbs_entry where room_id = '".$a."'
and (start_time between
'".mktime(01,00,00,date("m"),date("d"),date("Y")).."' and
'".mktime(24,59,59,date("m"),date("d"),date("Y")).."') order by id
Desc",$dbconn);

      if($result){
            $num = mysql_num_rows($result);
            $array = "";
```

```
            for($i=0;$i<$num;$i++)
            {
                  $row = mysql_fetch_array($result);
                  $array[$i]['id'] = $row['id'];
                  $array[$i]['start_time'] = $row['start_time'];
                  $array[$i]['end_time'] = $row['end_time'];
                  $array[$i]['name'] = $row['name'];
                  $array[$i]['file'] = $row['file'];
                  $array[$i]['link'] = $row['link'];


                  $array[$i]['timestamp'] = $row['timestamp'];


                  $array[$i]['link_type'] = $row['link_type'];

                  }



return $array;

}

 }

?>
```

D. clientview.php

 # clientview.php

```
<?php
  require_once 'lib/nusoap.php';
// request to webservice at its location
  $soap = new
nusoap_client('http://130.237.15.244/webservices/web_service.php?wsdl',
true);
  $proxy = $soap->getProxy();
  $doc = new DOMDocument();
// Based on the id of each display
  $doc->load( 'displayid.xml' );
  // fetch the data on the basis of tag and id
  $data = $doc->getElementsByTagName( "data" );
  foreach( $data as $record )
  {
      $ids = $record->getElementsByTagName( "id" );
      $id = $ids->item(0)->nodeValue;
  }

  $result = $proxy->add($information, 11);


// create doctype
$dom = new DOMDocument("1.0");

// display document in browser as plain text
// for readability purposes
//header("Content-Type: text/plain");


// create root element
$root = $dom->createElement("information");
$dom->appendChild($root);


for($i=0;$i<sizeof($result);$i++)
{
// create child element
      $record = $dom->createElement("record");

      $root->appendChild($record);

// create attribute node
      $id = $dom->createAttribute("id");
      $record->appendChild($id);

// create attribute value node
      $idValue = $dom->createTextNode($result[$i]["id"]);
      $id->appendChild($idValue);
```

```
        // create child element
        $start_time = $dom->createElement("start_time");
        $record->appendChild($start_time);

        // create text node
        $text = $dom->createTextNode($result[$i]["start_time"]);
        $start_time->appendChild($text);

         // create child element
        $end_time = $dom->createElement("end_time");
        $record->appendChild($end_time);

        // create text node
        $text = $dom->createTextNode($result[$i]["end_time"]);
        $end_time->appendChild($text);

        // create child element
        $name = $dom->createElement("name");
        $record->appendChild($name);

        // create text node
        $text = $dom->createTextNode($result[$i]["name"]);
        $name->appendChild($text);

        // create child element
        $link = $dom->createElement("link");
        $record->appendChild($link);

         // create text node
        $text = $dom->createTextNode($result[$i]["link"]);
        $link->appendChild($text);

        // create child element
        $file = $dom->createElement("file");
        $record->appendChild($file);
        $text = $dom->createTextNode($result[$i]["file"]);


        $file->appendChild($text);

         // create child element
        $link_type = $dom->createElement("link_type");
        $record->appendChild($link_type);

        // create text node
        $text = $dom->createTextNode($result[$i]["link_type"]);
         $link_type->appendChild($text);

        // create child element
         $link_type = $dom->createElement("timestamp");
        $record->appendChild($link_type);

         // create text node

         $text = $dom->createTextNode($result[$i]["timestamp"]);
                  $link_type->appendChild($text);


}
// save and display tree
 $dom->save("data.xml");
```

```
 echo "saved..";

?>
```

***********clientview.php code also include how information stored in their fields on client side***********

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="refresh" content="10" >
<title>Welcome to KTH @ Wireless</title>
<style type="text/css">

<!--
.style2 {
      font-size: 30px;
      font-weight: bold;
}
.style5 {font-size: 20px; font-weight: bold; }
.style6 {font-size: 10px; font-weight: bold; }
.style7 {
      font-family: Arial, Helvetica, sans-serif;
      font-weight: bold;
      color: #000099;
      font-style: italic;
}
.style9 {color: #CCCCCC; font-weight: bold; }
.style10 {font-size: 16px; font-weight: bold; }
.style11 {font-size: 16px}
.style12 {color: #000099}                                          c
.style13 {color: #000000}

-->
</style>

.style13 {color: #000000}
-->
</style>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"
/>
</head>
<body bgcolor="#c0c0c0">
<h1 align="center"><a href="http://www.kth.se"><img src="" border="0"
alt="KTH"/></h1></a>
<p align="center" class="style2">KTH@Wireless</p>
<p align="center" class="style5">Client Side Digital Display Information
System Software </p>
<p align="left" class="style10"> </p>
```

```
<p align="left" class="style12"><span class="style10"><span
class="style13">Display Name</span>: </span><span class="style11">Main
Entrence </span></p>
<table width="719" border="0" align="center" cellpadding="0"
cellspacing="6" id="Information" style="border-bottom-style:dashed,1px;">
  <caption align="top">
  <span class="style7">Information to be displayed on Digital Display
</span>
  </caption>
  <?
  $doc = new DOMDocument();
  $doc->load( 'data.xml' );

  $information = $doc->getElementsByTagName( "record" );
  ?>

  <tr style="background-color:#666666;">
    <td width="353"><div align="center"
class="style9">File/URL/Image/Video</div></td>

    <td width="101"><div align="center" class="style9">Start Time
</div></td>
    <td width="101"><div align="center" class="style9">End Time
</div></td>
    <td width="150"><div align="center" class="style9">Date& Entry Time
</div></td>
    <?

foreach( $information as $record )
  {
      $link_types = $record->getElementsByTagName( "link_type" );
      $link_type = $link_types->item(0)->nodeValue;

      $links = $record->getElementsByTagName( "link" );
      $link = $links->item(0)->nodeValue;
      $files = $record->getElementsByTagName( "file" );
      $file = $files->item(0)->nodeValue;
      $files = $record->getElementsByTagName( "file" );
      $file = $files->item(0)->nodeValue;

      $start_times = $record->getElementsByTagName( "start_time" );
      $start_time = $start_times->item(0)->nodeValue;

      $end_times = $record->getElementsByTagName( "end_time" );
      $end_time = $end_times->item(0)->nodeValue;

      $timestamps = $record->getElementsByTagName( "timestamp" );
      $timestamp = $timestamps->item(0)->nodeValue;
  ?>
  <tr>
    <td><? if($link_type == "1")
      echo $link;
      else echo $file; ?> </td>
```

```
    <td><?=date("g:i:s A",$start_time);?></td>
      <td><?=date("g:i:s A",$end_time);?></td>
    <td><?= $timestamp;?></td>
    <td> </td>
 </tr>
  <?
  }
  ?>
</table>
<p align="left" class="style6"> </p>
<p align="center" class="style5"><br />
</p>


</body>
</html>
</body>
</html>
```

E. Display.php

 # Display.php

```
<html>
<head>
<?php
  $doc = new DOMDocument();
  //  function to fetch the contents of the url to xml file
  $doc->load( 'data.xml' );
  // getting values from the elements inside
  $information = $doc->getElementsByTagName( "record" );
  $i = 0;
  $link_type[$i] = 1;
 // By default it will always display the main web page of KTH on the
first priority and if there is no information to show it will always show
this web page.
  $link[$i] = "www.kth.se";
  $i = $i + 1;
  foreach( $information as $record )
  {
  // To return all elements of the tag  in a tree
      $start_times = $record->getElementsByTagName( "start_time" );
      $start_time = $start_times->item(0)->nodeValue;

      $end_times = $record->getElementsByTagName( "end_time" );
      $end_time = $end_times->item(0)->nodeValue;

      $timestamps = $record->getElementsByTagName( "timestamp" );
      $timestamp = $timestamps->item(0)->nodeValue;
  // It will compare the current time function with the given time of the
content
      if(time() >= $start_time && time() <= $end_time)
      {
            $link_types = $record->getElementsByTagName( "link_type" );
            $link_type[$i] = $link_types->item(0)->nodeValue;
  // If the type of the information is 1 it will always display URL , but
if the type is 2 it will display a File( image)
            if($link_type[$i] == 1)
            {
                  $links = $record->getElementsByTagName( "link" );
            }
      $link[$i] = $files->item(0)->nodeValue;
            }
            $i = $i + 1;
      }
   }
  $size = sizeof($link);
// We defined a Count function, it will start from 0 and will continue to
display contents in a sequence, by default when count function becomes to
```

```
0 it will display the main webpage of kth.se and then follow the sequence
of the contents.
$count = 0;
if(empty($_GET['count']))
{
	$random = 0;
	$count = 1;
}
else
{
	$count = $_GET['count'];
	if(($count+1) < $size)
	{
		$random = $count;
		$count = $count + 1;
	}
	else if(($count+1) == $size)
	{
		$random = $count;
		$count = 0;
	}
	else if(($count+1) > $size)
	{
		$random = 0;
		$count = 1;
	}
}

// Meta function to refresh the page after 5 sec and also the location of
web service in the main server
echo "\n<meta http-equiv=\"refresh\"
content=\"5;http://130.237.15.244/webservices/display.php?count=".$count.
"\">\n";
?>

</head>
<?
if($size > 0)
{
if($link_type[$random] == 1)
{
?>
// we wil use the technique of frames to display the contents of the
WebPages
<frameset rows="*,*">
  <frame src="http://<?= $link[$random]?>">
<frame src="UntitledFrame-2"></frameset><noframes></noframes>
<?
}
else
{
	$arr = explode(".",$link[$random]);
```

```
        }
        else
        {
?>
// Set the location of our image file where it is stored
<img src="130.237.15.244/mrbs/files<?= $link[$random]?>" >
<?
        }
}
}
else
{
        echo "No Item Currently Available.";
}
?>
</html>
```

F. Display.C

 # The following code is generated in C i.e. to communicate with serial port of the display and turn off the display automatically at the end of the day.

```
#include <termios.h>

#include <stdio.h>

#include <unistd.h>

#include <string.h>

#include <stdlib.h>

#include <fcntl.h>

#include <sys/signal.h>

#include <sys/types.h>


#define BAUDRATE B38400

#define MODEMDEVICE "/dev/ttyS1"

//POSIX compliant source

#define _POSIX_SOURCE 1

#define FALSE 0

#define TRUE 1



char devicename[80]="ttyS0";

// derived baud rate from command line

long BAUD;

long DATABITS;

long STOPBITS;

long PARITYON;

long PARITY;

int Data_Bits = 8;              // Number of data bits

int Stop_Bits = 1;              // Number of stop bits

int Parity = 0;                 // Parity as follows:

                                // 00 = NONE, 01 = Odd, 02 = Even, 03 =
                                // Mark, 04 =Space
```

```
FILE *input;

FILE *output;

int status;


 main(int argc, char *argv[])

{


   int fd, tty, c, res, i, error;

   char In1;

   char str_command[9]="POWR0   ";

   struct termios oldtio, newtio;       //place for old and new port
settings for serial port

   struct termios oldkey, newkey;       //place tor old and new port
settings for keyboard teletype

   struct sigaction saio;               //definition of signal action

   char buf[255];                       //buffer for where data is put


   input = fopen("/dev/tty", "r");      //open the terminal keyboard

   output = fopen("/dev/tty", "w");     //open the terminal screen


   if (!input || !output)

   {

      fprintf(stderr, "Unable to open /dev/tty\n");

      exit(1);

   }


      error=0;

//set the user console port up


tty = open("/dev/tty", O_RDWR | O_NOCTTY | O_NONBLOCK);
tcgetattr(tty,&oldkey);
```

```
// save current port settings so commands are interpreted right for this
program

// set new port settings for non-canonical input processing must be
NOCTTY

      newkey.c_cflag = BAUDRATE | CRTSCTS | CS8 | CLOCAL | CREAD;

      newkey.c_iflag = IGNPAR;

      newkey.c_oflag = 0;

      newkey.c_lflag = 0;       //ICANON;

      newkey.c_cc[VMIN]=1;

      newkey.c_cc[VTIME]=0;

      tcflush(tty, TCIFLUSH);


      tcsetattr(tty,TCSANOW,&newkey);


        BAUD  = B9600;

        DATABITS = CS8;

        STOPBITS = 1;

        PARITYON = 0;

          PARITY = 0;


//open the device (com port) to be non-blocking (read will return
immediately)


        fd = open(devicename, O_RDWR | O_NOCTTY | O_NONBLOCK);

        if (fd < 0) {

         perror(devicename);

         exit(-1);

      }

      //install the serial handler before making the device asynchronous

      fcntl(fd, F_SETOWN, getpid());

      fcntl(fd, F_SETFL, FASYNC);


      tcgetattr(fd,&oldtio); // save current port settings
```

```
      newtio.c_cflag = BAUD | CRTSCTS | DATABITS | STOPBITS | PARITYON |
PARITY | CLOCAL | CREAD;

      newtio.c_iflag = IGNPAR;

      newtio.c_oflag = 0;

      newtio.c_lflag = 0;       //ICANON;

      newtio.c_cc[VMIN]=1;

      newtio.c_cc[VTIME]=0;

      tcflush(fd, TCIFLUSH);

      tcsetattr(fd,TCSANOW,&newtio);


      //Send Command to Turn off the Display


      write(fd,&str_command,strlen(str_command));


      tcsetattr(fd,TCSANOW,&oldtio);

      tcsetattr(tty,TCSANOW,&oldkey);

      close(tty);

      close(fd);        //close the com port


     fclose(input);

     fclose(output);

}    //end of main
```

TRITA-ICT-EX-2009:207

www.kth.se