FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO


FEUP

# Personal Health Channel

**Júlio Miguel Viana dos Santos**

Dissertation

Master in Informatics and Computing Engineering

Supervisor: Teresa Galvão (Ph.D)

Second Supervisor: Paula Silva (Ph.D)

Third Supervisor: Filipe Abrantes (Ph.D)

$28^{th}$ June, 2010

# Personal Health Channel

**Júlio Miguel Viana dos Santos**

Dissertation
Master in Informatics and Computing Engineering

Approved in oral examination by the committee:

Chair: Ana Cristina Ramada Paiva Pimenta (Ph.D.)
External Examiner: José Ângelo Braga de Vasconcelos (Ph.D.)
Supervisor: Maria Teresa Galvão Dias (Ph.D.)

---

$22^{nd}$ July, 2010

# Abstract

Europe's demographic pyramid is inverting at an ever-growing pace, bringing along drastic consequences to social security. The active workforce can no longer provide for the senior citizens. To revert this situation, which puts this increasing age group at risk, the social system must undergo a paradigm shift in order to cut costs and at the same time guarantee high standards of elder healthcare. Telemedicine is one of the most promising areas in this context.

This project's core is strongly connected to telemedicine, as it is integrated in the context of eCAALYX, a three-year project — April 2009 to April 2012 — funded by the European Commission under the Ambient Assisted Living Joint Programme. One of the eCAALYX components is a Personal Health Channel, a software application that enables users to manage their health and to streamline communication with the health services.

The main goal of this project is to perform a comparative study of the available technological tools to develop and deploy the Personal Health Channel using COTS solutions. This allows for a great reduction in efforts and costs of the project. An in-depth analysis of these tools will be performed in order to understand how they can be leveraged to pursue the project's goals.

# Resumo

A pirâmide demográfica da Europa tem vindo a inverter-se a um ritmo crescente. Tal inversão acarreta consequências drásticas para a segurança social. A força de trabalho activa já não pode providenciar o suficiente para manter os cidadãos sénior. De modo a inverter esta situação, que coloca este grupo demográfico crescente em risco, o sistema social deve sofrer uma mudança de paradigma no sentido de cortar custos, garantindo em simultâneo elevados padrões de cuidados de saúde à terceira idade. A telemedicina é uma das áreas mais promissoras neste contexto.

O núcleo deste projecto está fortemente ligado à telemedicina, visto que se integra no contexto do eCAALYX, um projecto de três anos — Abril de 2009 a Abril de 2012 — fundado pela European Commission sob o Ambient Assisted Living Joint Programme. Um dos componentes do eCAALYX é o Personal Health Channel, uma aplicação de software que permite que os utilizadores façam a gestão da sua saúde, e ajuda à comunicação com os serviços de saúde.

O objectivo principal deste projecto é efectuar um estudo comparativo das tecnologias disponíveis para desenvolver e distribuir o Personal Health Channel, usando soluções COTS. Isto permite uma grande redução de esforços e custos do projecto. Conduzir-se-à uma análise destas ferramentas para fomentar a compreensão de como estas podem ser usadas para cumprir os objectivos do projecto.

*"Tu, que te dizes Homem!*
*Tu, que te alfaiatas em modas*
*e fazes cartazes dos fatos que vestes*
*p'ra que se não vejam as nódoas de baixo!*
*Tu, qu'inventaste as Ciências e as Filosofias,*
*as Políticas, as Artes e as Leis,*
*e outros quebra-cabeças de sala*
*e outros dramas de grande espectáculo*
*Tu, que aperfeiçoas sabiamente a arte de matar.*
*Tu, que descobriste o cabo da Boa-Esperança*
*e o Caminho Marítimo da índia*
*e as duas Grandes Américas,*
*e que levaste a chatice a estas Terras*
*e que trouxeste de lá mais gente p'raqui*
*e qu'inda por cima cantaste estes Feitos...*
*Tu, qu'inventaste a chatice e o balão,*
*e que farto de te chateares no chão*
*te foste chatear no ar,*
*e qu'inda foste inventar submarinos*
*p'ra te chateares também por debaixo d'água,*
*Tu, que tens a mania das Invenções e das Descobertas*
*e que nunca descobriste que eras bruto,*
*e que nunca inventaste a maneira de o não seres*
*Tu consegues ser cada vez mais besta*
*e a este progresso chamas Civilização!"*

Almada Negreiros

# Contents

CONTENTS

# List of Figures

LIST OF FIGURES

# Abbreviations

| | |
|---|---|
| ASP | Active Server Pages |
| APE | Ajax Push Engine |
| API | Application programming interface |
| BAN | Body Area Network |
| CATV | Cable television |
| CE-HTML | Consumer Electronics HTML |
| CEA | Consumer Electronics Association |
| CGI | Common Gateway Interface |
| COTS | Commercial-off-the-shelf |
| CSS | Cascading Stylesheets |
| DOM | Document Object Model |
| HTML | Hypertext Markup Language |
| HTTP | Hypertext Transfer Protocol |
| IPTV | Internet Protocol Television |
| MCML | Media Center Markup Language |
| MVC | Model - View - Controller |
| NTSC | National Television System Committee |
| PAL | Phase Alternate Line |
| RTMFP | Real Time Media Flow Protocol |
| RTMP | Real Time Messaging Protocol |
| SATV | Satellite television |
| SDK | Software Development Kit |
| SIP | Session Initiation Protocol |
| SQL | Structure Querying Language |
| STB | Set-top box |
| TCP | Transmission Control Protocol |
| TV | Television |
| UDP | User Datagram Protocol |
| UHF | Ultra High Frequency |
| VHF | Very High Frequency |
| WBS | Wearable Body Sensor |
| WDK | Yahoo! Widget Development Kit |
| WMC | Windows Media Center |
| XBMC | Xbox Media Center |
| XML | Extensible Markup Language |
| YW | Yahoo! TV Widgets |

# Chapter 1

# Introduction

This chapter provides an introduction to the project at hand, Personal Health Channel. It starts with an overview of the context in which the project is integrated, and then it provides a more detailed explanation of the objectives of the project.

At the end of this chapter we can find an overview of the structure of the rest of the document.

## 1.1 Motivation and context

Europe's demographic pyramid is inverting at an ever-growing pace, bringing along drastic consequences to social security. The active workforce can no longer provide for the senior citizens [eca]. To revert this situation, which puts this increasing age group at risk, the social system must undergo a paradigm shift in order to cut costs and at the same time guarantee high standards of elder healthcare. Is it thus imperative to develop alternative solutions to reduce the resource consumption by the social system, while maintaining - or even improving - the life quality of the elderly. Telemedicine is one of the ways these goals can be achieved.

This project's core is strongly connected to telemedicine, as it is integrated in the context of eCAALYX, a three-year project – April 2009 to April 2012 – funded by the European Commission under the Ambient Assisted Living Joint Programme. The Project builds on the strengths of the infrastructure and functionality already developed in the previous CAALYX project [KBLA+09]. This project is being developed through established partnerships between several stakeholders, among which are Fraunhofer Portugal, Telefónica Investigación y Desarollo, INESC Porto, and several international hospitals.

eCAALYX aims to monitor elderly people with multiple chronic conditions, in order to prevent the deterioration of their health condition and improve their quality of life. The

output of the project is expected to be a solution commercially viable, acceptable by all users/stakeholders, reliable, long-term, flexible, scalable, and virtually maintenance-free in non-technical environments [Bou09].

In order to evaluate the developed system, and to enable its continuous improvement, eCAALYX will undergo two field testing iterations. The first one is due February 2011 and the second one February 2012. The system behavior and user feedback will provide valuable clues for perfecting the system. These testes will last from two to four weeks, and count with 20 patients from the cardiology department from the Charité hospital in Berlin, along with their health professionals.

This project was developed at Fraunhofer Portugal, a research institution which work is in the areas of research and development for assistive information and communication environments such as Ambient Assisted Living.

## 1.2 Project

### 1.2.1 Overview

This project is a subset of the aforementioned eCAALYX project which comprises a study of end-user's equipment to profit from the project. It consists of an analysis and selection of the best technologies to integrate a home display for Ambient Assisted Living in existing conmmercial-off-the-shelf (COTS, meaning technology already available for sale) solutions to implement a Personal Health Channel. This allows for less logistical planning and reduced costs. The existing hardware and software platforms will be compared and evaluated in order to conclude which ones are the best suited for the project.

After the aforementioned analysis is complete, this project will also include the conception, development and implementation of a prototype for the solution.

This project places a special focus on elderly users. As such, the system will follow an interaction model adapted to the senior citizens, taking into account the special limitations this age group has when interacting with a digital platform. Design guidelines for the television will be studied in order to ease elderly users into using the technology with less effort.

### 1.2.2 Objectives

The main goal of this project is to perform a comparative study of the available technological tools to develop and deploy the Personal Health Channel. An in-depth analysis of these tools will be performed in order to understand how they can be leveraged to pursue the project's goals. This will result in a deeper understanding of each tool, their advantages and weaknesses, and finally their fitness for the project.

One of the main criteria for the aforementioned selection and analysis has its roots on the Personal Health Channel higher priority features. These were carefully considered by Fraunhofer Portugal and are as follows.

**View health sensor data** The application will display a chart for each of the health signs being monitored.

**See health agenda** The application will feature the user's health agenda, which is a list of next medical appointments.

**Alerts for taking medication** The application will provide users with modal alerts to remind them to take their medication.

Another lower-priority feature will also be considered to build the scope of this work. This is the possibility of engaging in a videoconference with the medical services. This solution will not be implemented, but the requirements for the implementation will be studied.

The corresponding graphical interfaces were conceived in cooperation with another colleague at Fraunhofer Portugal.

## 1.3 Document structure

The next chapter will present the project in more detail, as well as its integration in the larger eCAALYX project. It will also feature an overview of telemedicine. Chapter 3 will discuss media consumption on television, present commercial offers aligned with the project's objectives, and provide an overview of telemedicine case studies. Chapter 4, the kernel of this work, features the choice of platforms and the thorough study of the chosen ones. Chapter 5 provides an evaluation of the prototypes built to gain additional insight on these technologies and the Personal Health Channel interfaces. Finally, chapter 6 will have the conclusions of this work.

# Chapter 2

# Problem description

This chapter provides a clearer formulation of the problem at hand. It starts by offering a more thorough explanation for the Personal Health Channel and how it is integrated in eCAALYX, and provides an overview of telemedicine, the yet broader context of this work.

## 2.1 Personal Health Channel

### 2.1.1 Overview

The Personal Health Channel concept is an application to assist its users to manage their health, providing them with a larger degree of independent living. Its graphical user interface will be available on a regular television screen, not only taking advantage of already existing display equipment, but also affirming itself as a centerpiece of the user's daily life. Interaction with the application is going to be provided through a remote control.

The Personal Health Channel is essentially a portal for users to have easy access to various health-related features. These features are still under consideration, but might include the following possibilities:

- access to historical charts which data come from health sensors;
- viewing of health-related videos recommended by the health services;
- making and receiving videoconference calls to doctors or the emergency services;
- checking the next medical appointments on the medical agenda;
- customizing the user interface and experience to one's own needs;
- receive alerts of next medical appointments or medication.

This study consists in the choice and analysis of the prime platforms available for the deployment of such a Personal Health Channel. It will begin by a market review, looking

for platforms which might be suitable. Then, a selection will be performed on these platforms, in order to select the most promising ones to achieve the project's goals. These chosen platforms will be studied in depth, to provide guidance for future development. Then, an evaluation study will be conducted in order to obtain further insight on the selected platforms.

This study will be based on three priority features and a secondary one. These are related to the health sensors, the medical agenda, the alerts system and videoconference. Videoconference will not be implemented on the aforementioned prototypes, but it will be thoroughly analyzed.

### 2.1.2 eCAALYX integration

This project is part of the much vaster eCAALYX project, mentioned on the previous chapter. The Personal Health Channel encompasses the TV and TVBox represented on figure 2.1. The system should be as less disruptive as possible for the user TV experience. This means it should minimize hardware requirements, and be compatible with the current TV viewing experience.

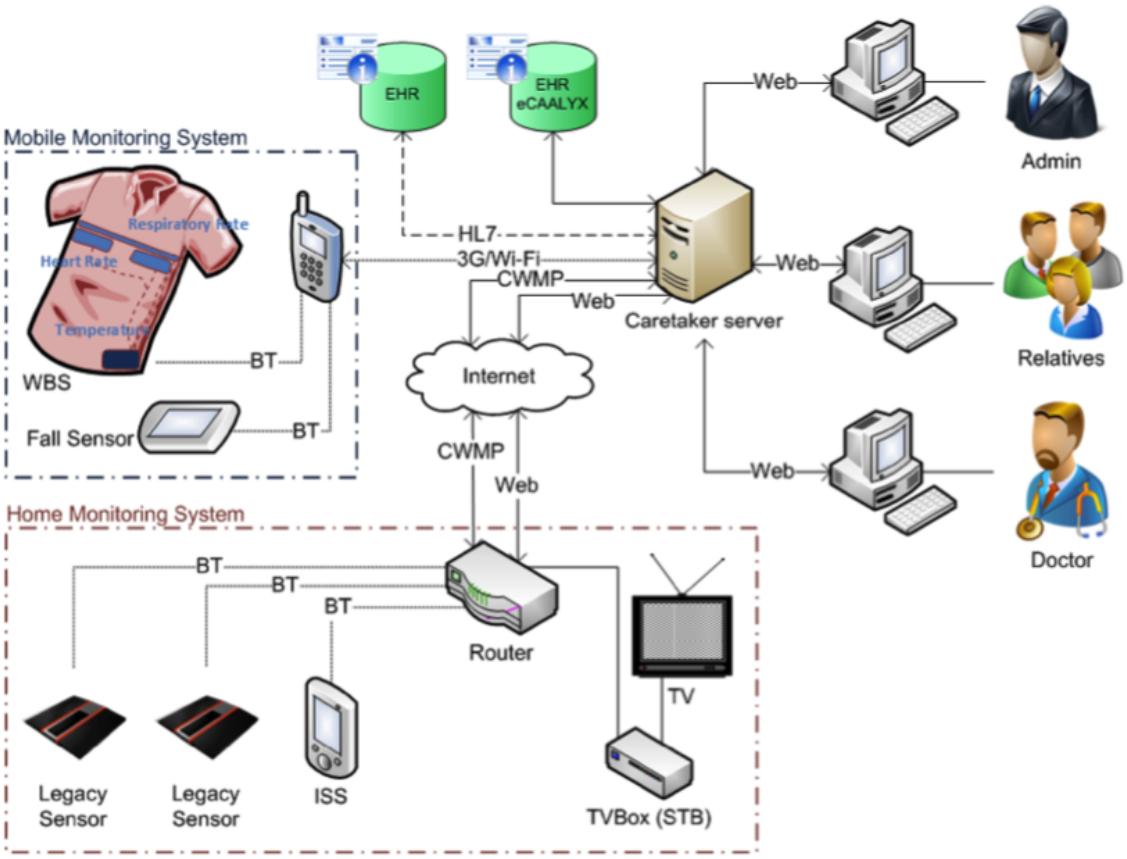


Figure 2.1: The eCAALYX project diagram.

The Personal Health Channel will be fed with data from the Caretaker server. These data are provided by the sensors on the Home Monitoring System and the Mobile Monitoring System. Some data will also be entered by the medical services, such as medical appointments, and medication prescriptions.

Since the eCAALYX project will have to undergo the first iteration of field testing on February 2011, this study also proposes a setup of the test beds, based on the study of the chosen platforms. This setup is presented on Appendix A.

### 2.1.3 Prototypes

In order to gain further insight on the selected platforms, a set of prototypes will be developed and tested. These prototypes will be based on a user interface study conducted simultaneously by a colleague at Fraunhofer Portugal.

## 2.2 Telemedicine

Europe's demographic pyramid is inverting at an ever-growing pace, bringing along drastic consequences to social security. The active workforce can no longer provide for the senior citizens [Bou09].

On the other hand, the lives of the elderly have become more active. They are not expected to be confined to any hospital or care-taking institution premises. Remote monitoring of patient status can help minimize house calls or trips to the hospital, while at the same time keeping a close eye on the patient's condition, alerting the health services for unusual health patterns. This allows for both expense cuts and an increase of the degree of freedom of the patient. Further exploration of telemedicine brings us to the use of videoconference for medical consults, which contributes to the previously stated argument [KBLA+09].

Is it thus imperative to embrace telemedicine as an alternative solution to reduce the resource consumption by the social system, while maintaining - or even improving - the life quality of the elderly.

In order to allow for a more personal contact between the patient and the carer, videoconference is a very desired feature in telemedicine deployment.

Videoconference support is usually provided either via computer software or dedicated hardware solutions. However, there have been some successful integrations of such a feature in set-top boxes, as described in [Tra03]. This integration comes across as seamless to the user, because it eliminates the need for a separate device.

### 2.2.1 Definition

Telemedicine (also known as Telehealth) has been defined as the use of telecommunications to provide medical information and services [Per95]. Such a broad definition circumscribes all kinds of possible uses, spanning from telephone advice to and from other medical professionals, the use of videoconference for remote consultations, or remote robot control for delicate surgeries. Another definition is the use of medical information exchanged from one site to another via electronic communications to improve, maintain, or assist patient's health status [Bro96].

### 2.2.2 Types of telemedicine

Telemedicine can either be practiced in a synchronous or asynchronous way. Synchronous, or real-time telemedicine demands the simultaneous presence of all the parts involved in the communication process. Telephone calls and videoconference fall into this category. Monitoring devices relaying their respective information to a remote site also constitute synchronous telemedicine [Bro96, Unk]. Asynchronous, or store-and-forward, telemedicine is an offline interaction. Regular postal mail is a form of asynchronous communication, as is the delayed assessment of collected information on a patient [Bro96, Unk].

### 2.2.3 Telemedicine issues

According to [Mil01], who focus on telemedicine as communication between doctor and patient, numerous telemedicine evaluation frameworks have been proposed. [Don80] distinguishes between technical and interpersonal components of such evaluations, in which the technical aspects are clinical diagnosis and outcomes, while interpersonal aspects are user satisfaction and acceptance. [PS95] found that patients are more willing to accept telemedicine after having had some experience with it than health care providers, who exhibit more resistance.

[RL.95, WR97] show that there is still no consensus on whether telemedicine enhance or damages the therapeutic relationship. There are still not enough studies to determine the effect of telemedicine on patients' communication of their discomfort, symptoms or socio-emotional state. The same applies to doctors' communication of treatment instructions or expressions of empathy and caring. [Mil01] also notes that because interpersonal communication between doctors and patients provides the basis for establishing comfort and trust, more research is needed if we are to guarantee that the appropriate conditions exist for exchanging information to be used to make health-care decisions. The extent of patient and doctor participation during the medical encounter can either shift towards

patient-centered and consumer.centered patterns, or reinforce traditional paternalistic patterns.

Privacy and security are also major concerns in telemedicine. [PBBL89] points out that privacy can be either informational, psychological, social and physical. The former refers to patients' personal files and data. Psychological privacy has to do with the revealing of intimate attitudes, beliefs and feelings. Social privacy means the ability to control social contacts. The latter refers to the ability to control physical accessibility. Telemedicine allows for the recording of medical consultations, which may rise privacy concerns on patients and doctors alike.

However, in a survey of the literature on telemedicine and doctor-patient communications, [Mil01] finds overwhelming evidence in favor of telemedicine, based on the participants' opinions and several other criteria. However, [HJ88] points out that patients tend to report high levels of satisfaction with the care they receive, which may influence telemedicine evaluation.

# Chapter 3

# State of the art

This chapter begins with relevant considerations on television and media consumption. It will then proceed with a survey of the available commercial products for media consumption. Finally, this chapter will present a set of telemedicine case studies and end with an overview of related work.

## 3.1 Television and Media consumption

Since the introduction of commercial broadcast television on the 1930s, television has earned a growing presence on people's homes. Now it is almost ubiquitous, being present on almost every living room on developed countries [tvp]. Some of these countries feature more than 1000 TV stations or channels, catering to every need and taste [tvc].

For many years, TV was broadcast through analog waves from the emitting stations to the consumer's houses. Today, TV content is often made available through subscription-based services broadcasted over cable (CATV), broadband Internet (IPTV), or satellite downlinks (SATV). These services are operated by multichannel video programming distributors [tvo] and require special hardware for signal decoding, grouped under the umbrella term *Set-top box*.

Television usage is subject to a different paradigm than the personal computer's, and should be regarded as such. These differences are often distinguished saying that TV viewing is performed on a *lean back* position, and PC usage on a *lean forward* one [wikg]. The context of TV is associated with leisure, while PCs often relate to work. The goals of TV are related to entertainment and relaxation, and PCs tend to emphasize productivity. The activities users engage when interacting with the TV are general free exploration, while on the PC they focus on task completion. Other relevant distinctions is the number of simultaneous users, which is usually one on the PC realm but can be multiple when

talking about TVs. Finally, TVs and PCs differ on their usual input devices: TVs are operated through a remote control, and PCs make use of a keyboards and a mouse.

### 3.1.1 Interactive television

This project involves a high degree of interactivity in order to accomplish its goals. End users must interact with the system to be able to reap their benefits. As such, interactive TV is a relevant concept for this study.

Interactive TV is a very broad term encompassing a number of methods that allow users to interact with television content. It enables simple operations such as adjusting the televisions's volume, and complex ones such as actively affecting the programs being watched [wikg].

Interaction can be achieved in a number of ways. It requires either a data return path, or special hardware designed to simulate interaction. To illustrate the latter case, one can imagine a sports transmission with several camera angles which is downloaded to the STB, allowing the user to select the viewing angle among the downloaded ones. This process is independent of the existence of a data return path, since there is no need to request additional content because the transmission itself already includes the several viewing angles.

If a return path is used, information can be sent back to the broadcaster. A return path implementation is not required to use the same transmission system as the broadcasting medium [wikg]. It can be either through telephone lines or mobile communications networks, for instance - this is a requirement when user's premises equipments have only receiving hardware, which is the case for SATV, for instance. CATV users can often use the same cable as a return path, and IPTV users use the underlying IP network for such.

### 3.1.2 CE-HTML and media convergence

Television media consumption's evolution is well illustrated by the emergence of standards such as CE-HTML, which shows the trend towards a connected, interactive television which takes on much more roles than those of simply broadcast TV display.

CE-HTML is part of the CEA-2014 standard by the Consumer Electronics Association [wikc]. The CEA-2014 standard defines the necessary mechanisms to allow a user interface to be remotely displayed on and controlled by devices or control points other than the one hosting the logic [CEA07]. It will allow consumers to control their CE devices from virtually anywhere, by providing a browser-based communications method for CE devices on a UPnP home network using Ethernet and a special version of HTML called CE-HTML [cea].

The CE-HTML specification — which is a subset of existing open Internet standards, such as XHTML1.0, CSS TV Profile 1.0, DOM level 2 (the Core, Style, Events, and

a subset of the HTML modules), ECMAScript 262 (3rd edition) and XMLHttpRequest (AJAX) — describes specific extensions for TV use, including Media object (for A/V rendering control), spatial navigation (up-down/left-right/OK) and guidance for text input with a simple TV remote control (SMS style) [phi].

This standard is important because if all TV manufacturers adopt the same standard, one website will be interpretable by all TV sets, irrespective of brand. This will limit unnecessary diversity and promote a much higher level of efficiency for content and service providers, enabling them to offer more services to a larger homogeneous installed base [phi].

### 3.1.3 Set-top boxes

*Set-top box* (STB) is an umbrella term used to encompass a multitude of different devices which ultimately connect to a display such as a television (TV). These range from simple TV tuners to advanced computers providing an interactive television user experience [itv, wikt].

Set-top box is not a scientific term. This name comes from its original form factor: a box which stood on top of the users's television [wikt].

A STB can be seen as an information appliance which takes an external source of signal, converts it to a suitable format for display, sending this translated form to the display device (a TV in most cases) [wikt].

Usually the external signal is encoded, in order to minimize storage and network bandwidth requirements. As such, STBs commonly feature decoders. Decoders are hardware or software solutions to revert the encoding performed on data, changing it back to the original format. Another common requirement is the presence of local storage buffers, accounting for the mitigation of network jitters to assure consistent playback [itv].

The signal source might be an ethernet cable, a satellite dish, a coaxial cable, a telephone line, power line, or a VHF/UHF antenna. Depending on the medium used for transmission and whether the STB allows uplink connections, the signal content may take multiple forms, ranging from simple audio/video (AV) to interactive webpages or games [itv, wikt].

As STB complexity increases, it allows the implementation of more features. A simple broadcast STB with no return path (uplink connection) merely allows for content consumption. Should it integrate one, its user will be able to access an extended service array, such as Video on Demand, e-commerce, or Internet-related amenities including email, chat, and more [itv].

Contemporary STBs are endowed with good processors, memory, middleware, software applications and even hard disk drives. They uncover a more advanced and demanding set of services, such as high-speed Internet access, high-definition interactive

TV, digital video and audio recording, time-shifting and gaming [itv].

### 3.1.4 Personal computers

A Personal Computer (PC) is a computer suitable for an individual to use, not only because it supports his or hers most common operational requirements, but also features an appropriate form factor while staying price-wise accessible.

Television content can be replaced or complemented with Media Center (MC) functionalities provided by a PC. It can, for instance, enable users to browse and play either local (e.g. on a hard disk drive) or remote (e.g. on a network-attached storage device) audiovisual media and display it on their television. It might also support Internet TV, videoconference, email, games, and other entertainment features.

## 3.2 Commercial products for media consumption

The market offers a vast array of consumer products and services for media consumption. This section provides an overview of selected set of commercial offers for media consumption deemed relevant in the context of this project. These offers will later undergo a selection process and some of them will be used for this project.

### 3.2.1 OEM STB Software

In this section, two platform solutions for STBs are analyzed. These are Microsoft Mediaroom and OpenTV Core, common solutions used by service providers to embed on commercial set-top boxes.

#### 3.2.1.1 Microsoft Mediaroom

Microsoft Mediaroom (MM) is Microsoft's solution for harnessing the power of IPTV, whether it be live or on-demand television. It is the latest update of the Microsoft TV IPTV Edition platform software, which runs on a set-top box connected to a Microsoft IPTV network. It supports on-demand and live video, digital video recording, time shifting, and interactive program guide with integrated search and scheduled recording [wikj, med].

Application development of the MM platform uses the Microsoft Mediaroom Presentation Framework, which is based on the ASP.Net framework [med].

Further details can only be obtained through a development partnership with Microsoft. Members of the Microsoft Mediaroom Application Development Program are sponsored by their service provider customers and work directly with these service providers to create the latest content and applications [med].

#### 3.2.1.2 OpenTV Core

OpenTV Core (OTVC) is a digital television middleware for set-top boxes. It features support for on-demand and live high-definition television, digital video recording and time shifting [wikn, opea].

Applications are written in C with OpenTV-proprietary libraries, the OpenTV SDK; however, Adobe Flash Lite is currently being studied as an alternative [wikn, ado]. SDK Development Suite C2.2 is the software kit that enables to write, compile, build, test and debug interactive applications using OpenTV Core2.2 [opeb].

### 3.2.2 Limited STBs

This section presents STBs that don't allow for live broadcast TV play back. These will be called *limited STBs*.

#### 3.2.2.1 Neuros LINK

The Neuros LINK, from Neuros, is a dedicated computer running the XBMC media center. It can play all major local media content, as well as Internet TV. Since it is a Linux setup, it has the ability to run several hundred applications built for this operating system from its user interface [neu, wikl].

#### 3.2.2.2 Boxee Box

The Boxee Box, built by D-Link, runs the Boxee Media Center on a linux operating system. It features a remote control and a remote QUERTY keyboard. The user can access Internet TV and play all major local media content [wikb, boxa].

Application development is possible through Python plug-ins to the Boxee Media Center [wikb, boxb].

The Boxee Box is still under development, and is expected to be available this year [boxa].

#### 3.2.2.3 Popbox

Very similar to the Boxee Box, this device from Syabas Technology runs the David Box platform and also features a remote control. Development for such platform is conducted on the Adobe Flash Lite technology [pop].

The Popbox is still under development, and is expected to be available Spring 2010 [pop].

#### 3.2.2.4 Myka ION

Myka ION is a limited STB capable of playing Internet television and local media. It comes pre-installed with XBMC Media Center, Boxee, and Hulu Desktop [mykb].

Its development program hasn't started yet, but applications can be built for the installed media centers [myka].

### 3.2.3 Interactive TV platforms

Recently, some major TV vendors have been integrating ethernet interfaces on their TV sets, in order to enable otherwise unavailable interactive features. Some of these systems are open, i.e. the vendors make SDKs accessible for developers to build applications to feature alongside theirs.

#### 3.2.3.1 Yahoo! TV Widgets

Yahoo TV Widgets are rich Internet applications designed to run on the Yahoo! Widget Engine platform. This platform is available in a variety of consumer electronic devices, including flat TV panels from Samsung, Sony, LG Electronics and VIZIO. The TVs run Linux and use a specially modified version of the Yahoo! Widget Engine, an application platform derived from their Konfabulator desktop widget platform. [yw:a].

Widgets are available through the Yahoo! Widget Channel, which can be accessed on the TV and used to download widgets that have been approved by Yahoo! [yw:f].

#### 3.2.3.2 Opera Widgets

Opera Widgets technology enables small, often single-purpose, applications built using open Web technologies to provide useful information or services to end-users. The entire Web application is delivered (by download, push, or pre-installation) and can run locally on virtually any device like a native application without launching a browser or directly accessing the Web [oped].

To distribute widgets directly to consumers that use Opera Software products on their devices, the widget simply needs to be uploaded to the Opera Software's official widgets gallery. Once the widget has gone through an approval process, it is made available for download [opec].

Access to the Opera TV Widgets SDK is restricted and, despite all efforts conducted, Opera refused access at this stage.

#### 3.2.3.3 Philips NetTV

Philips NetTV, unlike Yahoo! TV Widgets and Opera Widgets, is not widget-based. It features a full CE-HTML browser, able to access selected web applications developed by Philips partners [phi].

Access to the Philips NetTV partnership program, required for development, is restricted and, despite all efforts conducted, Philips refused access at this stage.

#### 3.2.3.4 JavaFX TV

JavaFX is a Java platform for creating and delivering rich Internet applications that can run across a wide variety of devices. JavaFX 1.3 features support for the development of TV applications, including a new TV emulator [jav]. TVs and other devices with an Internet connection running the Java Virtual Machine should be able to run JavaFX applications, which are widgets, not unlike Yahoo!'s or Opera's.

At the time of this writing, no TVs supported JavaFX.

#### 3.2.3.5 Google TV

Google TV is a software platform for set-top boxes and HDTVs based on the Android operating system and co-developed by Google, Intel, Sony and Logitech [wikf].

Current Android applications will be supported on the TV. Google TV runs the Chrome web browser and it is capable to use the Flash Player plug-in, enabling most websites to work on Google TV [goo].

Development for Google TV is so far unfeasible, since it is limited to Google partners. The SDK add-on with Google TV-specific extensions will be available a few months after first product availability [goo].

### 3.2.4 PC media centers

In this section we will analyze three current MC solutions: Windows Media Center, XBMC + MythTV, and Boxee.

#### 3.2.4.1 Windows Media Center

Windows Media Center (WMC) is a MC application designed by Microsoft and prepared to run on the latest Microsoft Windows platforms.

Using WMC, the user can view the most common media types on his or hers television, whether they are pictures, videos, or music. Also, if the PC in which WMC is installed is equipped with a TV tuner, WMC can play back and record Broadcast TV. Finally, whenever an Internet connection is available, WMC allows for Internet TV viewing [wmcd].

Although proprietary, WMC is extensible, allowing developers to create new applications to integrate with it. Applications are developed using the Microsoft .NET Framework 2.0, the WMC API, and XML-based Media Center Markup Language to display the user interface [wmce].

#### 3.2.4.2 XBMC Media Center and MythTV

XMBC Media Center (XBMC) is a free, open source and cross-platform media center, with a strong emphasis on its front-end interface. It serves as an alternative for Windows Media Center, except for its native inability to play back and record broadcast television [wikz, xbm]. However, it can communicate with a MythTV (an open-source third party free Unix application) backend, which eliminates what would otherwise be such a shortcoming [myt, wikk].

XBMC allows developers to use the Python programming language to create new applications [wikz, xbm]. MythTV is open and written in C++ [myt, wikk].

#### 3.2.4.3 Boxee

Boxee is a cross-platform freeware derivative from the XBMC Media Center, unable to play back live broadcast TV [boxc]. Its strong suit is its social media component, which closely integrates social media websites with the media center activity.

Development options are provided in the same way as for XBMC [wikb, boxb].

#### 3.2.4.4 Web browsers

Web browsers are ubiquitous software application on the PC medium. The most important mainstream offers – Safari, Chrome, Firefox, Opera and Internet Explorer – are very mature and capable. They can also be extensible using plugins. Web browsers can be leveraged in order to serve as media centers, since they provide access to local media.

#### 3.2.4.5 Custom software

Because of the open nature of a PC and the most common operating systems, software can be built from scratch to meet all the user's needs.

## 3.3 Telemedicine case studies

In this section we present five projects related to telemedicine, in an effort to provide a broader overview on the subject. These projects are: CAALYX, eCAALYX (which constitutes the context for this work), CogKnow, EasyLine+, I2HOME and Intel Health Care Management Suite.

### 3.3.1 CAALYX

The goal of CAALYX is to permanently monitor the health status of elderly users for the purpose of predicting/detecting any unfolding adverse health conditions and preventing complications before they develop, while respecting user's privacy and personal life needs. If an adverse condition arises, the system relays a high priority message to an emergency service (e.g., 112), including the geographic position and clinical condition of the elder user [Bou09].

The system is composed of the following subsystems:

**Mobile Monitoring Subsystem** Collects and monitors key vital signs and detects adverse health events and falls when users are outdoors; facilitates efficient communications between the user and his/her caretakers and family in the advent of an emergency when the elder person is away from the home environment.

**Home Monitoring Subsystem** Monitors users while at home and also helps keep them in touch with their family and caretakers; delivers classic services (television, videophone) and features rich Internet communications.

**Caretaker's Monitoring Subsystem** Provides concurrent monitoring of a number of elders by specialized personnel in an efficient manner. The caretaker decides whether to promote a raised event to the emergency service (112) or not.

### 3.3.2 eCAALYX

eCAALYX is the follow-up to the previously discussed CAALYX project. Its objectives can be summarized as follows [Bou09]:

- Health monitoring of older and elderly persons with multiple chronic conditions, at home and on the move (the original CAALYX did not cover the health monitoring and management of older people with comorbidity).
- Improve the quality of life of elderly persons by increasing their freedom and safety. Prevent deterioration of the patient condition by providing continuous support, guidance, and relevant health education.
- Achieve all of the above goals by providing a solution that is commercially viable, acceptable by all users/stakeholders, reliable, long-term, flexible, scalable, and virtually maintenance-free in non-technical environments, thus suitable for real-world deployment.

eCAALYX is composed of three main subsystems:

- The Home Subsystem, which includes Customer Premises Equipment (CPE), Set-top-box (STB)/interactive TV (to deliver health education and other functions), Tricorder and home sensors (those sensors that are stationary and not continuously worn on the body), all of them located at home;
- The Mobile Subsystem, which includes a smart garment, with all sensors integrated into a wireless BAN—Wearable Body Sensors (WBS), and a mobile phone;
- The Caretaker Site, which includes the Caretaker Server and the Auto-configuration Server.

### 3.3.3 CogKnow

The objective of the project is to develop a user-validated, cognitive prosthetic device and associated services for elderly people with mild dementia. Such a solution will have a tremendous impact on the quality of life and autonomy of persons with mild dementia and will potentially increase the period of time they can remain independent and living at their own home [Bou09].

Its main technical components are:

- The Home-based hub, responsible for the collection of all information pertaining to the activities of the person suffering from dementia within his/her home, and for relaying this information to the CogKnow Web Server. It is a stationary device located at a fixed position within the person‘s home.
- The Mobile Cognitive Prosthetic, which purpose is to mirror the services offered by the Home-base Hub, so that they may also be accessible from anywhere.
- The Web-based Server, which acts as a repository of the entire system, allowing the carer to configure and schedule patient reminders.
- The Sensorised Home Environment, a set of sensors and actuators attached to domestic appliances. These communicate with the Home-based Hub.

### 3.3.4 EasyLine+

Six partners from three European countries have developed prototypes suitable for the market of advanced white goods to support elderly people with or without disabilities in carrying out a longer independent life at home, and to compensate for any loss of physical and/or cognitive abilities they might have in the kitchen. This is achieved by bringing ambient intelligence to the kitchen environment [Bou09].

Instead of relying on new smart appliances with accessible interfaces, EasyLine+ features a central intelligence system that is aware of the status of all white goods in the

kitchen, able to control them, and also able to interact with the user. This essentially means giving home appliances the capacity to communicate. A Human-Machine Interface (HMI) is managing user interaction and is of key importance to the system [Bou09].

### 3.3.5 I2HOME

The I2HOME project builds on two main pillars [Bou09]:

- The implementation of the ISO/IEC 24752 Universal Remote Console (URC) standard;
- The development of user interfaces for disabled persons following a user-centered design methodology.

I2HOME followed a user-centered design methodology to develop four distinct user interface prototypes for the cognitively disabled, Alzheimer‘s patients, the elderly, and sensory impaired patients. I2HOME‘s ultimate goal is to provide a unified and accessible user interface to common home appliances and consumer electronics based on industry standards. This approach should make it much easier for people with cognitive disabilities and older persons to operate these devices, e.g., by using an intuitive interface running on an Apple iPhone, without having to learn and remember a different and complex operating interface for each appliance separately, thus helping these persons live independently at their own homes for longer periods of time [Bou09].

### 3.3.6 Intel Health Care Management Suite

The Intel Health Care Management Suite is a web-based application for healthcare professionals to connect with their patients using the Intel Health Guide PHS6000. The Suite provides healthcare professionals a set of tools to manage their patients, including [Gro09a]:

- Communication tools, such as video conferencing.
- Tools to develop personalized care protocols based on an individual’s health needs.
- Tools to prioritize patients’ clinical needs based on their reported status.
- Access to comprehensive patient-specific data.

The Intel Health Guide PHS6000 is a personal health system which combines an in-home patient device with an online interface, allowing clinicians to monitor patients and remotely manage care. It enables patients to [Gro09b]:

- Participate in their own care by monitoring their health status under the guidance of a healthcare professional

- Communicate with healthcare professionals
- Learn about their health and condition

## 3.4 Related work

Work closely resembling this study was not found during its conduction. The reason for this fact might be that studies in the field are encumbered by a classified status, such as this one. There is arguably plenty of similar research conducted by health providers, television broadcasters and distributors, and even consumer electronics developers. All these constitute interested stakeholders involved on the value chain of such a project and, as such, it is unlikely that this study pioneers the field.

# Chapter 4

# Platform analysis

This chapter constitutes the kernel of this study. It starts with an informed decision on which platforms to further study for development, given those listed on the previous chapter. The reasons for such a selection will be detailed on section 4.1.

This chapter will also feature a technologically agnostic architecture proposal for the context of this project's immediate scope, which is the TV and connected devices. This should allow for a better comprehension of the project itself, and serve as reference for future development.

The results of the study of the chosen platforms will then be presented. For each one, this chapter features a technological background and development considerations.

Finally, this chapter presents further considerations on videoconference and push technology.

## 4.1 Choosing platforms

From the large number of platforms presented in chapter 3, only a few were chosen for in-depth study. A number of factors contributed to this triage. They are as follows.

**Time** The present work was conducted through a 4 month-period. This short timespan forced the trimming of the list of selected platforms.

**Availability** Some of the platforms are, due to various reasons detailed in this section, unavailable for development.

**Technology** The list of selected platforms was built considering only the most promising technologies, that is, those which are more aligned with the project's objectives.

### 4.1.1 The Microsoft Mediaroom conundrum

Shortly after the initiation of this research work, Microsoft Mediaroom was deemed to be the prime deployment platform given the intended project goals. It is a widely deployed interactive television solution built to harness the power of IPTV. Mediaroom is available for a large number of users across Europe, since is the middleware choice of many European IPTV providers.

Further details can only be obtained through a development partnership with Microsoft. Members of the Microsoft Mediaroom Application Development Program are sponsored by their service provider customers and work directly with these service providers. A deal with a portuguese IPTV provider is underway. However, this process's intrinsicalities make its establishment a slow operation, rendering the study of the platform unfeasible given this work's deadlines.

Testimonies from experienced Mediaroom developers in a portuguese IPTV provider, along with the exploration of deployed Mediaroom implementations – such as Portugal Telecom's Meo IPTV service – easily illustrate this platform's potential and adequacy. Henceforth, it is still a valuable candidate for the deployment of the Personal Health Channel, albeit its exclusion from the present study.

### 4.1.2 Other excluded platforms

#### 4.1.2.1 Commercial and legal issues

Additional hurdles emerged along the development of this study. Two other considered platforms were eventually excluded for reasons akin to Microsoft Mediaroom's – Opera Widgets and Philips NetTV. Access to the Opera TV Widgets SDK is restricted and, despite all efforts conducted, Opera refused access at this stage. In a similar fashion, access to the Philips NetTV partnership program, required for development, is restricted and, despite all efforts conducted, Philips also refused access at this stage.

Neither of these platforms was deemed as promising as Microsoft Mediaroom. However, they made interesting case studies due to their advertised standards-based implementation mindset.

Opera Widgets has its strengths in its common development framework, easing deployment on several devices supporting this technology.

Philips NetTV, however, appears to be an ad-hoc value-boosting marketing solution. As the name implies, its availability is limited to Philips TV sets, greatly reducing its ubiquity wherewithal and raising switching costs due to vendor lock-ins.

Further hindrances disqualified other platforms from supplementary analysis. Java FX TV, Google TV, the Boxee Box, Popbox and the Myka ION all suffer from diverse sorts of unavailability issues. Java FX TV, at the date of this writing, has not announce any

compatible platforms to the author's knowledge. Google TV, the Boxee Box and Popbox are still under development and its release is scheduled for later this year. Finally, the Myka ION Internet-based forums show a number of complaints from unmet deliveries upon payment, and it was eventually decided not to pursue development until the product is considered ready for production.

#### 4.1.2.2 Technological issues

In addition to the previously enumerated platforms, this study will also not consider three others for further analysis, due to technological issues: OpenTV Core from the middleware category, and XBMC plus MythTV, the Neuros LINK, and Boxee from the PC media center category.

OpenTV core, like Mediarrom, is a widely deployed STB middleware. However, it pales in comparison to Microsoft's solution. Firstly, an overview of the SDK denounces its development tools to be much less evolved than Mediaroom's. Additionally, its graphic libraries appear inflexible in comparison to the Microsoft solution[1].

The combination of XBMC and MythTV was also chosen to be left out of this study. It can be seen as an alternative solution to Windows Media Center, given its features and their potential. However, due to time constraints, further investigations were pursued only on the latter. There are two main reasons for this decision. The first reason is that Windows Media Center is developed by Microsoft, which was considered more trustworthy where system stability and technical support are concerned. The second reason is the fact that Windows Media Center is readily available on most versions of the Windows platform. The same case could be made for Neuros LINK: since it runs the XBMC Media Center, it was considered preferable to use Windows Media Center installed on a dedicated PC.

Finally, this study will not include the Boxee platform, since it's essentially a less flexible media center than Windows Media Center or XBMC. It does have interesting social features, but these were considered to be of little relevance to this project.

### 4.1.3 Custom software development

This study found that the available market offerings sufficed to accomplish the project's goals. Profiting from available frameworks and their features was deemed less time-consuming than developing a custom software solution from scratch. This option was nevertheless, if only briefly, considered at the start of this study.

A personal computer could be used as an additional device to be installed between the STB and the television. It would capture the decoded image signal from whichever STB the user owns, and superimpose the developed custom software solution over it. This way,

[1] OpenTV has recently released OpenTV SDK Development Suite C2.2 BETA, which was not analyzed.

the system would work along with any STB, without the need for software or hardware updates. Furthermore, the development environment would suffer from no restrictions whatsoever, because it wouldn't be bound to any platform.

This solution does, however, present some drawbacks. One comes from the fact that it does not offer a way to control the STB itself. Broadcast TV playback, for instance, would be affected, since the application would not be able to access playback controls. This could be, however, easily overcome if the CEA-2014 standard becomes ubiquitous on consumer electronics. Another problem is the need for the acquisition and installation of additional dedicated hardware, further encumbering the user not only financially, but also technologically.

### 4.1.4 The selected platforms

Three platforms were selected to be studied in further depth: Windows Media Center, Yahoo! TV Widgets and web browsers.

One of the reasons that lead to the aforementioned selection was the breadth of concepts it encompasses. Each platform belongs to a different class. Windows Media Center is a media center application for PCs running the Windows operating system. Yahoo! TV Widgets is an interactive television platform embedded on television silicon, transparentizing the hardware layer. Web browsers are tools used to navigate the World Wide Web.

Windows Media Center is an easily deployable solution. It comes pre-installed on most Windows versions, and an application can be installed with a simple Windows installer. Live broadcast TV viewing is a somewhat more complicated matter, requiring the installation of a TV tuner and configuration of Windows Media Center.

Yahoo! TV Widgets is a good example of an integrated solution, which favors uncluttering by embedding its hardware inside televisions. It is available on a considerable number of TV models from several vendors, unlike other solutions previously presented which offered a lesser degree of availability.

Web browsers have made their way from personal computers into a number of devices today, including televisions and set-top boxes. Their adequacy for this project heavily depends on the browser implementation and the features it supports, especially with concern to the Flash Player plugin. Most PC browsers offer support for all features, and there is a trend towards providing greater support on TVs and STBs.

## 4.2 Architecture proposal

In this section we present a back-end solution interoperable with the chosen platforms. This will ensure the correct publishing and distribution of data to users.

### 4.2.1 Overview

This project's scope is limited to the customer's premises equipment, and even there to the set-top box or an equivalent device. However, a suggestion for the immediate back-end of the services will be presented. This can be seen as a guide to future implementations, to ensure they are interoperable with this solution proposal.

The Personal Health Channel must be fed with data to accomplish its goal. We will assume the existence of an Internet server which holds these data, along with a specific architecture for its processing and delivery. Data are bound to the project objectives, mentioned in chapter 1.

All applications, regardless of the platform used for their development and deployment, must have access to the server so the relevant data are able to be pulled from the server, or pushed by the server to the application (see 4.5 for details on push technology).

### 4.2.2 Data

Chapter 1 lists three priority features which imply the existence of data to work properly.

Whatever the data source, it should be published to the server, and stored in a database. These should be made available through dynamic server-side scripts visible to the outside (such as PHP or CGIs) for a pull model. In the case of a push model, where the server sends unrequested data to clients, these data should be scheduled to be sent.

The first feature, *View health sensor data*, assumes the existence of health sensors able to publish their readings to a network. Each of these sensors has a particular function, and the data they emanate are thus diverse, albeit obeying to the same protocol and hence having the same format. The data might be formed as a list of pairs, a timestamp for the reading, and a list of one or more associated values: `{timestamp, {value}*}`.

The second feature, *See health agenda*, assumes that medical appointments have been set up by the health services and published to the server. Each appointment could be represented by a structure with information of the time and place of the appointment, the attending doctor: `{time, doctor, place}`, in which `time` could be `{date, time}`, `doctor` could be `{name, specialty, photo}` and `place` could be `{name, address, photo}`.

Finally, *Alerts for taking medication* assumed that a medication schedule has been set up for the user by the medical services. This schedule should be heavily customizable in order to allow the medical services to personalize it for each user's specific medication needs. The data structure could either be:

- a list of times for each specific medication: `{medication name, {time, dosage}*}`, in which `time` could be `date, time`;
- a frequency indicator for each specific medication: `{medication, dosage, frequency}`;
- a list of medications per time: `{time, {medication, dosage}*}`.

Note that the data structure examples will differ according to whether they are used for publishing or retrieval. In the first case, *View health sensor data*, publishing the sensor's data would require additional parameters, e.g. the user id and sensor type for correct data handling by the server.

### 4.2.3 Data delivery schedule

When server push systems are in order, it is important to take into consideration the data delivery schedule. Health support systems are critical systems, which should be planned and developed in a way that accounts for multiple malfunctions. One of the possible hurdles that has to be overcome is the eventuality of a prolonged loss of Internet connection.

In the case of pull systems, in which the clients poll the server from time to time for more information, this is not such a serious issue since the server will reply with the data as soon as it is polled. However, in opposite push systems, there is the possibility of sending relevant information to the clients only when the correct time of that information arrives. If the Internet connection is down at this time, the client would be deprived of potentially vital information. In order to overcome this flaw, push systems should deliver data as soon as possible, and let the client system handle its presentation when the right time arrives.

### 4.2.4 System architecture

In order to correctly publish, store and distribute information to benefit the users, a proposal for a system architecture to handle the data needs of this project follows. It assumes that the publishing endpoint is a browser or other Internet-enabled software application, to be used by the medical services. The data retrieval or receiving endpoints are the various platforms selected earlier in this chapter, on section 4.1.4.

This architecture supports both push and pull models for data distribution, although the push model – in which the server pushes the information to the client – is illustrated only by web browsers as receiving endpoints.

The presented system architecture consisted of several modules, each responsible for a a specific function. An overview of their operation, and how they fit into the global architecture, follows.

This architecture is technologically agnostic, and could be implemented in a number of ways.

- The *Main publishing entry point* and the *Data retrieval facility* could use PHP, Python or other dynamic page languages.
- The *Database* could be MySQL, Microsoft SQL Server, or other database management system.

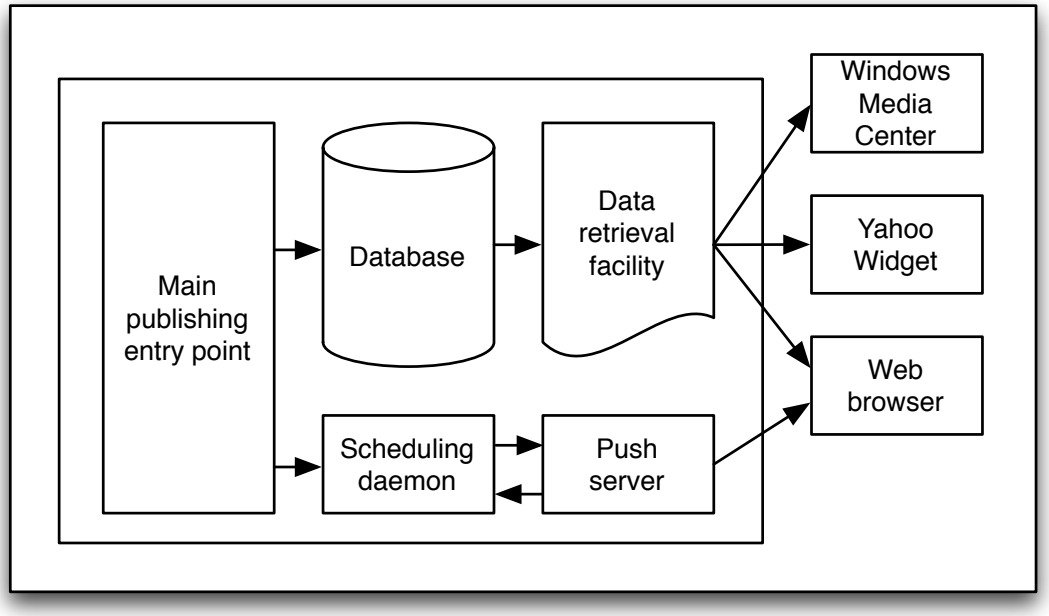


Figure 4.1: System architecture proposal.

- The *Scheduling daemon* could either be a custom software component, or Unix's *at daemon* working closely with a *scheduling manager* to handle rescheduling due to errors such as network malfunctions.
- The *Push server* could be an APE server, as discussed on section 4.5, or other Comet implementation.
- The *Data retrieval facility* could output either raw text or format it according to a markup language such as XML.

Selecting the appropriate technologies to better address issues such as server load expectations, memory usage, or caching, is outside the scope of this work.

**Main publishing entry point** This is the entry point for publishing data, and is accessed by the medical services. Its most important features are data validation, database access and push handling. The data validation function must ensure the data entered by the medical services is in the correct format for distribution. It should also check for duplicate information, inexistent addressees, and other inconsistencies. Database access is responsible for updating the database with the new data. Push handling is responsible for initiating the push process, notifying the *scheduling daemon* of a new scheduling rule.

**Database** The database is where all published data are stored for posterior access.

**Scheduling daemon** This module is responsible for scheduling data push to clients. It is generic enough to work while encompassing or not the considerations of section 4.2.3: This module can either wait for the scheduled time to notify the client, or push the data as soon as they arrive. Communication failures should be accounted for, and data pushing should be eventually retried if they happen.

**Push server** This module is responsible for pushing the data to the clients. If a failure to communicate occurs, the *Scheduling daemon* must be notified so it can retry later.

**Data retrieval facility** This module's function is to dynamically generate data pages based on the user requesting said data. When a client application makes a request for data, this module queries the database for whichever information was requested and outputs the data to a page readable by the application. This module should also account for user authentication.

## 4.3 Deployment Platforms

### 4.3.1 Windows Media Center

Windows Media Center (WMC) is a media center application designed by Microsoft and is prepared to run on the latest Microsoft Windows platforms. It is included in Windows XP Media Center Edition, premium editions of Windows Vista (Vista Home Premium and Vista Ultimate), and Windows 7 (Home Premium, Professional, Enterprise, and Ultimate) [wikx]. It is designed with 10-foot user interface guidelines, and it is usually initiated and controlled using a remote featuring the green media center button [wmcc]. Using WMC, the user can view the most common media types on his or hers television, whether they are pictures, videos, or music. Also, if the PC in which WMC is installed is equipped with a TV tuner, WMC can play back and record Broadcast TV. Finally, whenever an Internet connection is available, WMC allows for Internet TV viewing.

#### 4.3.1.1 Technological background

Although proprietary, WMC is extensible, allowing developers to create new applications to integrate with it. Microsoft makes a Windows Media Center Software Development Kit available, easing the learning curve by providing documentation and sample code that developers can use to create software. Applications are developed using the Microsoft .NET Framework 2.0, and XML-based Media Center Markup Language to layout the user interface [wmce].

The Windows Media Center Platform consists of two parts: the managed code object model and the Windows Media Center Presentation Layer.

The Windows Media Center managed code object model can be regarded as an API, since it's a collection of namespaces for controlling Windows Media Center features. It enables the developer to programmatically automate features and experiences of Windows Media Center, including media playback, queue management, tuning to live TV shows, scheduling recordings of future TV shows, parental controls, and navigation to Windows Media Center features from within a third-party application.

The Windows Media Center Presentation Layer separates an application's visual presentation from its logic by employing a model/view approach. The model provides the logic of the application (the code and data), and is developed using a managed code language, such as C#. The model is non-visual. The view provides the look and behavior of the application (the user interface), and is authored in MCML. MCML provides dynamic layout capabilities, integrated animation support, rich text and graphics support, automatic keyboard and remote navigation, parameterization, private local storage, conditional-based data binding, and access to managed code assemblies from markup [wmcb].

A Windows Media Center application is a managed Microsoft .NET Framework assembly that runs inside Windows Media Center [wmce]. We say that it is managed since it requires and will only execute under the management of a Common Language Runtime virtual machine [wiki].

Using the .NET Framework, developers may extend the functionality of Windows Media Center by using the programming interfaces exposed by the Windows Media Center application object model, the .NET Framework's System namespace, and namespaces that are provided by external assemblies. Using the Windows Media Center Presentation Layer allows applications to have access to the same rendering technologies that are used to create Windows Media Center itself [wmce].

.NET Framework assemblies are the building blocks of .NET Framework applications: they form the fundamental unit of deployment, version control, reuse, activation scoping, and security permissions. An assembly is a collection of types and resources that are built to work together and form a logical unit of functionality. An assembly provides the common language runtime with the information it needs to be aware of type implementations [dotb].

Assemblies are designed to simplify application deployment and to solve versioning problems that can occur with component-based applications. Many deployment problems have been solved by the use of assemblies in the .NET Framework. Because they are self-describing components that have no dependencies on registry entries, assemblies enable zero-impact application installation. They also simplify uninstalling and replicating applications [dota].

There are three types of Windows Media Center applications:

- A local application, consisting of an installed managed-code assembly and related files. As a locally-installed application, it has access to all computer resources.
- A web application, that uses MCML delivered using the HTTP protocol over the Internet to display rich client UI on the local PC without requiring installation of local files or assemblies. Windows Media Center provides an environment in which the MCML can be hosted and provides full access to Windows Media Center API methods and properties and a subset of Microsoft .NET Framework types to support common scenarios of web-delivered experiences. A Windows Media Center web application has no access to local computer resources.
- A background application, which has no user interface, launches soon after Windows Media Center is started, and continues to run until it closes on its own or is forced to close by Windows Media Center. A Windows Media Center background application can run continuously and transcend individual Windows Media Center experiences and features [wmca].

Application distribution is free and up to the developer. The developer may pack his application on a .msi installer file for the user or OEM to run in order to install the application on his media center.

#### 4.3.1.2 Application interface

Windows Media Center is designed to be started using the corresponding button on the remote control. It brings up a fullscreen window with the main menu (see figure 4.2). Background applications are loaded with Windows Media Center, running as soon as WMC initialization is complete. They don't have a graphical interface, but can launch either other applications or native WMC features.

Local WMC applications are accessible by default through the extra option on the main menu (see figure 4.3). It takes us to a list of the installed applications, where we can select which one we want to run.

If an application is opened, it will load its registered entry point, taking us to its main graphical interface (figure 4.4 is a WMC sample application from Microsoft called *Z*). Its appearance and behavior is entirely up to the developer, since no mandatory design guidelines apply.

#### 4.3.1.3 Development tools

The development environment for WMC, recommended by Microsoft, is the following setup:

- Windows 7;

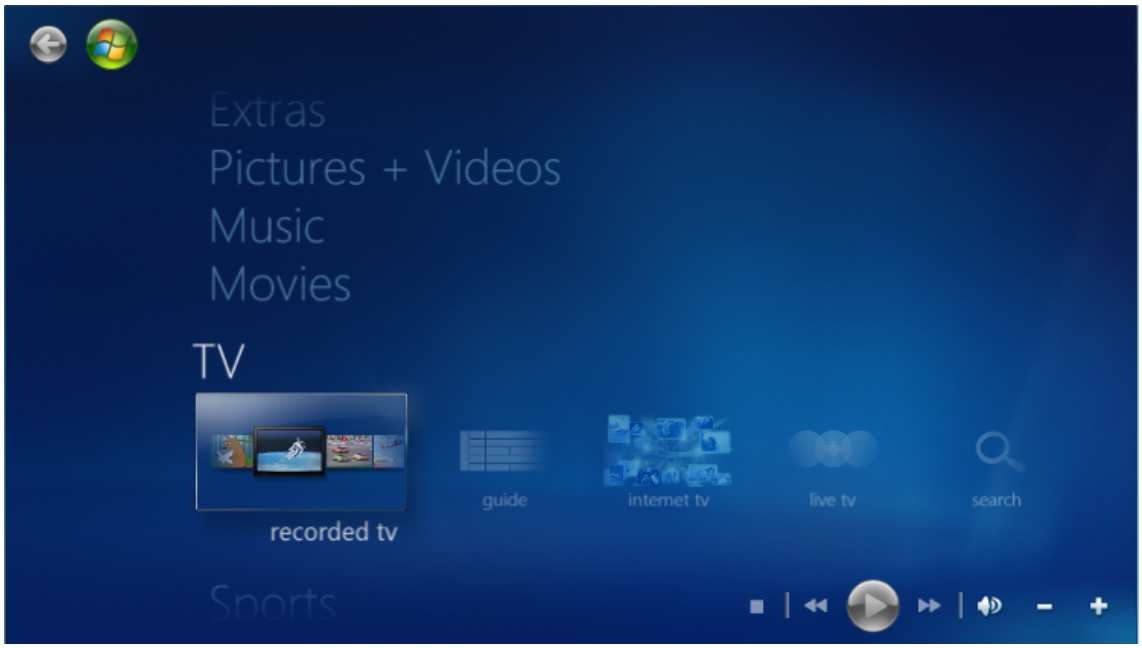


Figure 4.2: Windows Media Center main menu

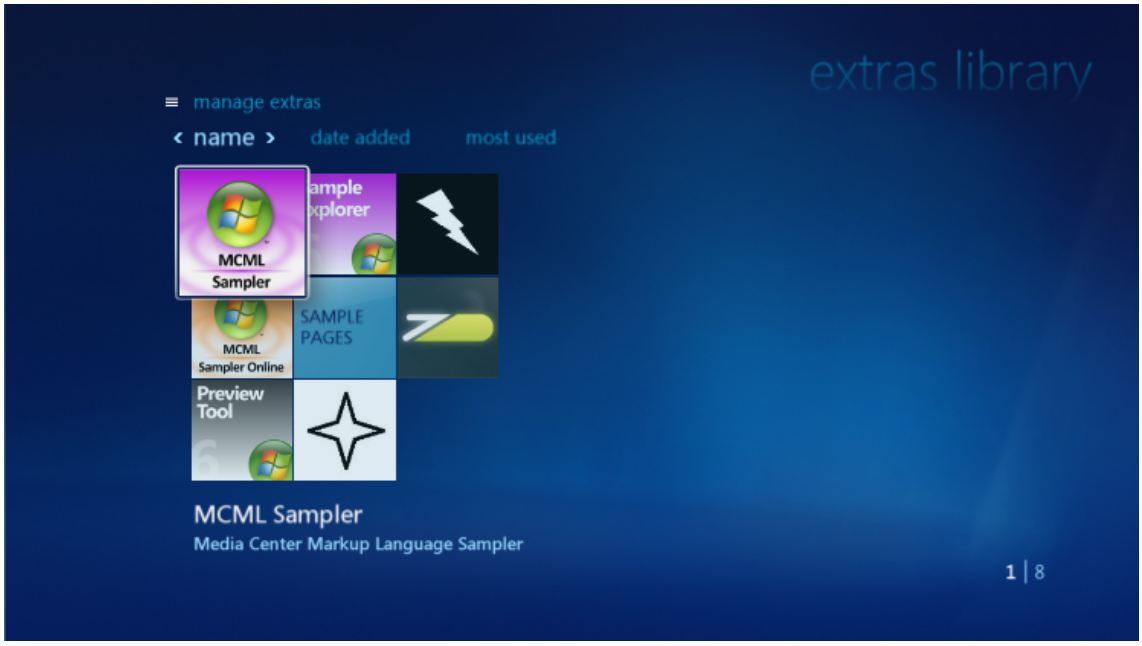


Figure 4.3: Windows Media Center extras menu

- Microsoft Visual C# 2008 Express Edition;
- .NET Framework 2.0;
- Windows Media Center Software Development Kit.

It is possible to use the full version of Microsoft Visual Studio, and other programming language such as Visual Basic instead of C#. However, the code samples and project

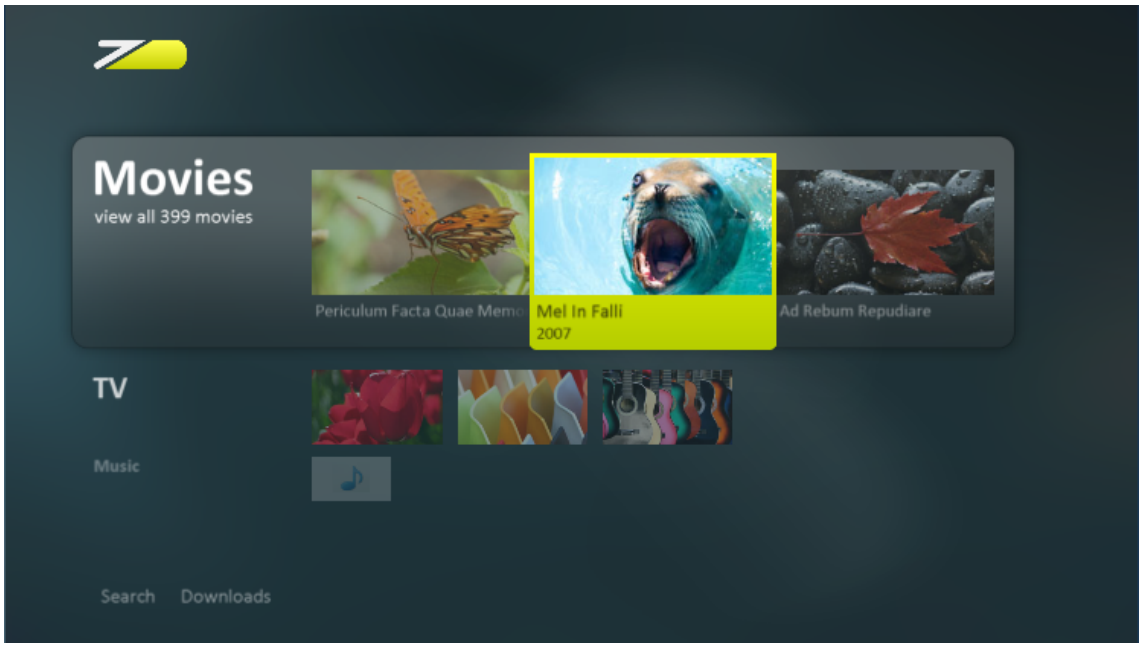


Figure 4.4: Windows Media Center sample application

templates available on the SDK are C#-oriented.

The SDK, freely distributed by Microsoft, is designed to help developers create applications and software components that take advantage of features provided by WMC and includes documentation, tools, sample code, and Visual Studio project templates.

#### 4.3.1.4 Development process

Developing WMC applications, either background or foreground, is simplified by using the SDK's Visual Studio project templates. They provide a code and markup scaffold which may then be built upon to fit the developer's needs.

The business logic of a WCM application rests within the C# code. These files comprehend the business logic, information and visual flow, and high-level application behavior. Through an MVC viewpoint, C# could be regarded as the both the controller and the model of the application.

The user interface is constructed using MCML. These compositions rest in separate files, defining the visual aspects and lower-level application behaviors such as button triggers. Developers who are new to WMC should be aware that MCML has a very steep learning curve and, without any WYSIWYG editor available, it becomes rather difficult to tackle. These files define the view, through an MVC viewpoint.

The relationship between the controllers and the views is pretty straightforward. The controller may load a certain view with any number of configuration properties, and other relevant data, should the view be subject to parameters. The view can also load additional information from the model through an optional referenced passed to it on initialization.

The view defines triggers to activate controller functions, for instance through button events.

Microsoft suggests several ways to test WMC applications. It is, however, recommended that users install the application in WMC for testing purposes, since this will provide the most accurate graphic behavior. Nevertheless, this is not an elementary process. In order to be installed, a WMC application must undergo a series of prior preparations, since only deployment-ready applications are allowed to be installed. This process needs to be done only once, and is well documented on the developer guide. The tools involved are also readily available. Testing is then achieved by running an installation script and reopening the WMC application one wishes to test.

#### 4.3.1.5 Special considerations

WMC specifics require additional clarifications in the matters of the requirements related to alerts and videoconference.

In order to issue alerts to the user independently of his or hers current activity, the alerts feature must be implemented outside the scope of the main application, in order to function even when the user is not using the main application. This raises the need to install a background application to handle such context-independent notifications. The implemented solution consisted in a background application which started along with WMC startup. This background application features a polling cycle in which it queries a remote server from time to time for alert information.

Videoconference is made possible by encoding local audiovisual capture media and publishing it to a remote server. This is attained by having the WMC application starting Windows Media Encoder on a console application on the background, in a way that is transparent to the user. Windows Media Encoder then handles capture and publishing the media according to a previously constructed session file which is passed to it as a parameter.

### 4.3.2 Yahoo! TV Widgets

Yahoo TV Widgets are rich Internet applications designed to run on the Yahoo! Widget Engine platform. This platform is available in a variety of consumer electronic devices, including flat TV panels from major vendors. The TVs run Linux and use a specially modified version of the Yahoo! Widget Engine, an application platform derived from their Konfabulator desktop widget platform. Yahoo has further extended and simplified widget development by building a JavaScript framework which abstracts most of the complexity from Konfabulator [yw:a].

#### 4.3.2.1 Technological background

TV Widgets can be developed by OEM's and 3rd party developers using the Yahoo! TV Widget KONtx Framework which provides component user interface elements and behaviors [yw:d]. Developers may get the Widget Development KIT, which includes:

- A PC Simulator, which allows developers to test their applications without resorting to an actual device.
- The developer guide, with an introduction to widget development and the API documentation.
- Code samples.

This is still a recent technology (2009), which makes the platform somewhat buggy, as we can see when browsing the Yahoo Connected TV Forum, the main source of information for developers [yw:b]. The platform's abilities are very limited, due to a lowest common denominator approach on consumer electronic devices' computing performance. Yahoo has a minimum device profile for devices running the platform, but it is more of a feature list than a document of minimum requirements. It deals with RAM amount, graphic resolution, and audiovisual format decoding [yw:c].

Widgets are made up of multiple files which are stored in a subdirectory of the Widgets directory in the production environment. The widget engine loads a file named `main.TV` which includes the application code for the widget [yw:e].

Widgets are built using the Yahoo! Widget Toolbox. The Yahoo! Widget Toolbox contains the common user interface elements and behaviors used in building widgets. The purpose of the Yahoo! Widget Toolbox is to:

- Help streamline the widget development process.
- Encapsulate primitive objects to provide extended functionality.
- Provide a normalized programming interface with consistency in widget layout and behaviors.
- Promote the adoption of Yahoo! design guidelines by widget developers.

The elements in the Yahoo! Widget Toolbox are the basic building blocks for widgets and encapsulate DOM nodes. The controls in the Yahoo! Widget Toolbox are extensions of the toolbox elements, combining basic elements, controls, and themes to provide ready-to-use user interface components.

The Yahoo! Widget Toolbox provides a Framework for the widget developer to build upon. All communication with the widget engine is wrapped by the Framework to simplify the development process.

A separate file encapsulates all the widget's metadata called `widget.xml`. This file includes information such as the widget's name, unique identifier, author, description, etc. It also specifies security settings for your widget. The security element controls access to resources such as the filesystem, http, command, and more.

In order for a widget to be available for download through Yahoo!'s Widget Gallery, it must comply with a series of both performance and aesthetic guidelines. Although only sparsely detailed, Yahoo! recommends developers abide to the default graphical design settings, to ease the acceptance process. Additionally, a set of general but strict design guidelines are provided [yw:e].

#### 4.3.2.2 Application interface

Yahoo Widgets are started by using the corresponding button on the remote control. This brings up a widget dock with snippets (see figure 4.5). Snippets are the tiles that are graphically displayed on the bottom of the television screen. Its content may be dynamically altered, and if the purpose of the widget is merely to display information (such as a stock price) a snippet will suffice.

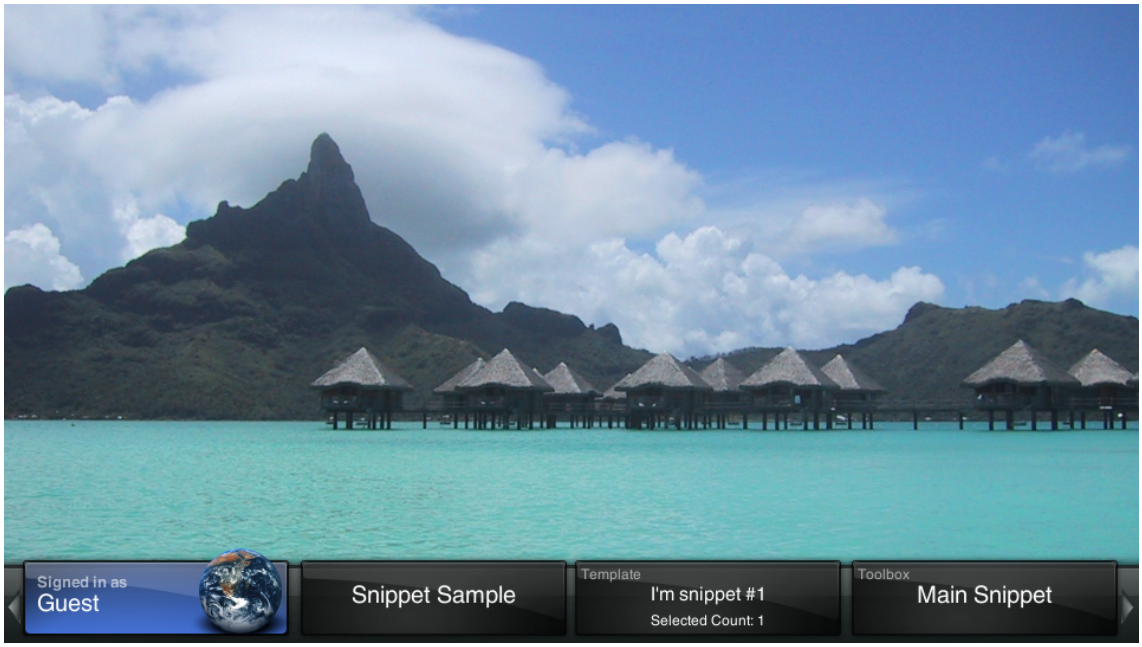


Figure 4.5: Yahoo! Widgets dock

The sidebar view (see figure 4.6) is the primary view of a widget. When a snippet is activated, it launches the widget from the dock into the widget's sidebar view.

As the widget grows in complexity, the developer may choose to endow it with a fullscreen view, accessible only through the sidebar.

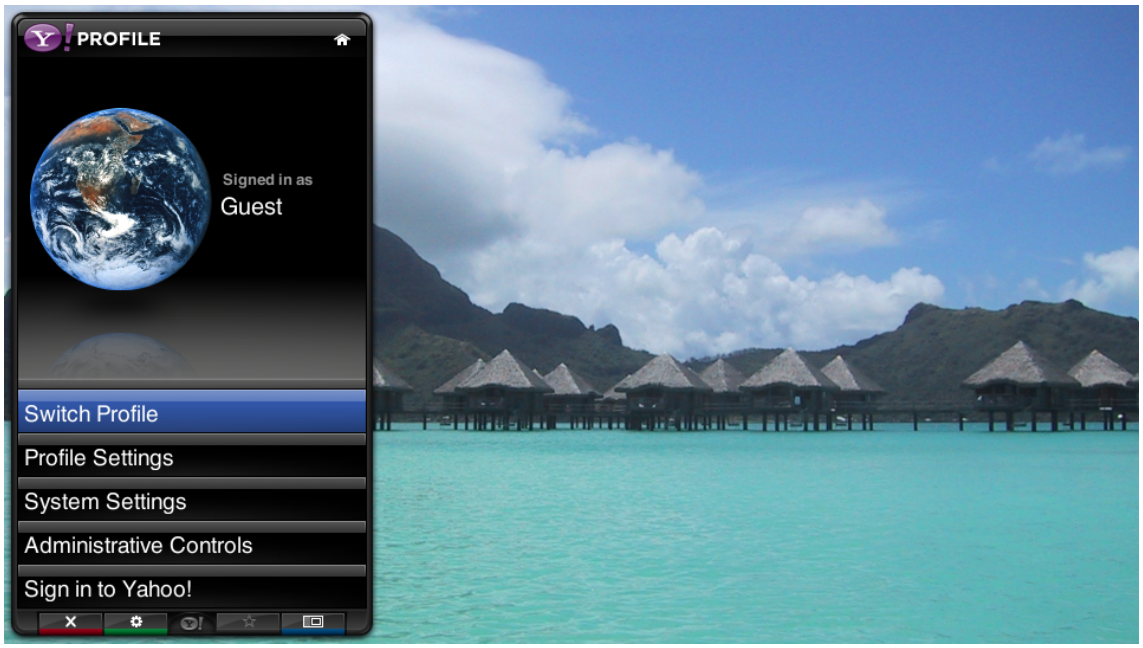


Figure 4.6: Yahoo! Widgets sidebar

#### 4.3.2.3 Development tools

In order to enable development of Yahoo! TV Widgets, the following software applications are required:

- a text editor suitable for JavaScript, XML and CSS;
- an image editor for creating JPG or PNG images;
- the Yahoo! Widget Engine Widget Development Kit (WDK).

The WDK includes a PC simulator (eliminating the need to resort to an actual compatible television), the developer guide, and code samples. The PC simulator is provided through a Debian software package, thus running on the Debian or Ubuntu Linux distributions. Yahoo! recommends using Ubuntu 8.0.4 LTS.

#### 4.3.2.4 Development process

Yahoo! suggests taking a sample widget and start development from there. Yahoo!'s sample widgets already possess the required deployment folder structure, and initialization and configuration files.

The development process is JavaScript-centered, aided by the use of Yahoo!'s proprietary KONTx framework, which help leverage this platform's specific capabilities. The developer guide is somewhat abridged, and developers will probably feel the need to browse the platform's Internet forums for additional information. These forums provide

decent moderated support and are the main source of information aside from the developer guide provided in the SDK.

Interface construction is centered on using and customizing the default UI components suck as dialogs, buttons and grids. The framework promotes the atomization of complete screens on a single JavaScript file, which defines behaviors for creating and updating said screen, as well as navigation and other triggers.

Applications may be tested on the Simulator included on the SDK. It is intended to simulate actual deployment of a widget on a compatible TV, and runs alongside a console with platform event logs, which can be used for debugging. This simulator is a rather underdeveloped system, which may prove to be somewhat buggy or unresponsive at times, and developers are advised to use an actual compatible TV for testing purposes. Yahoo! recommends acquiring all compatible TV models in order to ensure maximum widget compatibility.

#### 4.3.2.5 Special considerations

Alert handling on this platform is only possible inside the scope of the installed widget, since there is no support for background applications. Inside the scope of a widget, it becomes a trivial matter accomplished by implementing a timer on startup which triggers a server query and then possibly an alert event to notify the user. The further away developers can get from inside a widget is its respective snippet, which can be altered dynamically to feature a warning sign such as a conspicuous badge, but this will only be seen when the user explicitly enters the widget dock.

### 4.3.3 Web browsers

A web browser is a software application that enables the user to retrieve, present, and traverse information resources on the World Wide Web, namely web pages [wikv]. The evolution of web browsers, both in number and diversity, along with the growth of the set of available features, has been notorious over the past few years. However, for the scope of this document, the most relevant features of a web browser are:

- the layout engine;
- the scripting engine;
- the plug-in support.

#### 4.3.3.1 Layout engine

A web browser layout engine is a software component that takes marked up content and formatting information and displays the formatted content on the screen [wikw]. Marked

up content is content expressed in a markup language such as HTML. This provides the content on a hierarchical node tree. Formatting information takes usually the form of a style sheet. CSS is the common standard for web pages.

#### 4.3.3.2 Scripting engine

Most browsers support JavaScript as a client-side scripting language [wike]. JavaScript is an implementation of the ECMAScript language standard and its main usage on web pages is to interact with the DOM model and respective formatting information, dynamically altering its contents.

JavaScript support allows for the use of Ajax. Ajax is a group of interrelated web development techniques used on the client-side to create interactive web applications. With Ajax, web applications can retrieve data from the server asynchronously in the background without interfering with the display and behavior of the existing page [wika]. These data can then be used to edit the web page's DOM or formatting information.

#### 4.3.3.3 Plug-in support

Many browsers have support for plug-ins, which are software components built to extend the features of their host programs. Only the Flash Player plugin should be of concern in this work. This plugin enables the developer to embed Flash content on webpages. Flash technology is the mainstream provider of animations, movies, and streaming media online.

#### 4.3.3.4 Development tools

Given the multitude of web browser implementations available, there isn't a single responsible corps which to refer when recommending development tools. However, web browser development is relatively straightforward and ubiquitous. A possible configuration of the development setup is as follows:

- a text editor suitable for HTML, JavaScript and CSS;
- the web browser versions used for deployment.

Since there are many different web browser versions, it is recommended that development occurs on the one or ones on which the final release will be available, to ensure that no compatibility issues arise.

Videoconference development requires an additional set of tools, since it makes use of Flash. Henceforth, the previous list must be engorged with a flash authoring integrated development environment, such as Adobe Flash or Adobe Flex. Flex is freely distributed by Adobe, and the Adobe Flash graphical authoring capabilities, absent on Flex, are not

relevant for this project. Furthermore, Adobe's resources relates to Stratus are Flex resources. As such, Flex is recommended for development.

#### 4.3.3.5 Development process

Front-end web development is executed using HTML, JavaScript and CSS. HTML, the markup language, provides the semantic contents of the page and builds the Document Object Model. CSS are stylesheets used to control the presentation of the elements defined in HTML. JavaScript is used to dynamically manipulate DOM elements and stylesheets, along with implementing client-side business logic and, in the case of this implementation, establishing and managing the connection to the ajax push server. It is also used to install and manage triggers and events.

When compared to the other two aforementioned platforms, web development offers a much vaster array of possibilities considering application business logic. This is mostly implemented on the server who provides the web pages, albeit conceding some control to JavaScript when client-side business logic is deemed relevant – for instance, to validate input data before sending them to the server.

General purpose web applications should be tested on all major web browsers, user share-wise, to ensure maximum compatibility. They tend to implement certain aspects of CSS and JavaScript differently, and fine tuning is usually required to guarantee the same presentation in all of them.

#### 4.3.3.6 Special considerations

Alerts on the browser have been implemented resorting to an Ajax push server. If the implementation features URL navigation, developers should implement a connection to said server with each page the browser loads, since the connection won't be kept alive when a different page is loaded by the browser.

## 4.4 Videoconference

Videoconference is a set of interactive telecommunication technologies which allow two or more locations to interact via two-way video and audio transmissions simultaneously [wiku]. In order to make videoconference possible, each participant's endpoint must be equipped with video and audio capturing and outputting devices, as well as a data transfer device (usually an Internet connection).

Videoconference is achieved by efficiently streaming both inbound and outbound audiovisual media, which usually requires a server medium through which to set up the connection between endpoints.

### 4.4.1 Technological background

This section contains detailed explanations of videoconference possible implementations on the previously listed deployment platforms.

#### 4.4.1.1 Windows Media Center

In order to set up videoconference with Windows Media Center, a media encoder software service must be running so that it captures the image and video streams from the audio-visual input devices and publishes it to a remote media server. Microsoft already has a solution stack developed to make this possible: Windows Media Encoder and Windows Media Services.

Windows Media Encoder is a freely downloadable media encoder developed by Microsoft which enables content developers to convert or capture both live and prerecorded audio, video, and computer screen images to Windows Media formats for live delivery [wiky]. It can run on the background as a command-line application which takes a encoding session description file as a parameter. If so instructed, it is able to push stream the encoded video to a remote media server, like Windows Media Services.

Windows Media Services is a streaming media server developed by Microsoft. It is available as a free download for Windows Server, and has a number of advantages over other streaming media services when accesses by Windows Media clients. It allows for streaming speeds faster than the media data rate, forward error correction, and other benefits that add to the quality os service provided. Streaming can be achieved over either TCP or UDP.

Since Windows Media Center natively supports playing of streaming video files through a given URI, the ability to receive remote video streams from a videoconference session is guaranteed.

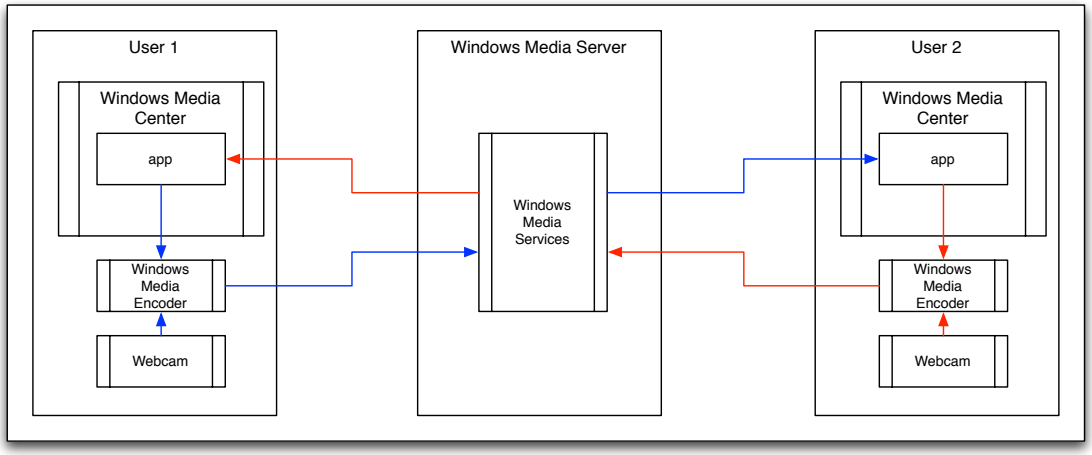


Figure 4.7: Windows Media Center videoconference architecture

#### 4.4.1.2 Yahoo TV Widgets

The Yahoo TV Widgets Engine does not expose system resources to developers. Hence, it is not possible to access audiovisual input devices in order to push them online. This makes videoconferencing impossible on this platform, even though it can play inbound streaming video.

#### 4.4.1.3 Web browsers

HTML and JavaScript are insufficient to accomplish videoconference in the browser, since they do not expose the host's system resources to developers. This has traditionally been circumvented using the Flash Player plugin, since it has access to the input devices required for videoconference. Flash can then publish the media to a remote server using RTMP, or set up a P2P connection between the videoconference users via a Stratus server using RTMFP.

**RTMP** RTMP is a proprietary protocol developed by Adobe Systems for streaming audio, video and data over the Internet, between a Flash player and a server [wikr]. It can either be used raw on top of TCP, encapsulated within HTTP request to traverse firewalls by connecting to port 80 (RTMPT), and even transmitted over a secure HTTPS connection (RTMPS). Flash captures media from the input devices and may push publish it to a remote Flash Media server via RTMP. This can be either Adobe's proprietary Flash Media Server, or a free, open-source solution like Red5. The current worked explored only the latter, hoping to benefit from reduced costs. Red5 is a free, open source Java implementation of a Flash Media Server based on reverse engineering of RTMP and AMF protocols. Like Flash Media Server, it supports streaming and recording of audio/video, live stream publishing, and Flash remote objects [wiks]. Red5 can be installed on a remote server and be expected to handle streaming without any configuration. Streams are published along with an arbitrary identifier and can be retrieved by making a request to Red5 asking for the stream carrying that identifier. The retrieved stream can then be displayed on the Flash Player plugin.

**RTMFP** RTMFP is a very recent communication protocol from Adobe that enables direct end user to end user peering (P2P) communication between multiple clients running an application built for Adobe Flash Player for the delivery real-time communication. RTMFP uses P2P techniques to ensure a high quality delivery and efficient use of the network. It is a managed connection, which means it requires the authorization of a Stratus server to make the introductions. [str]. Stratus is a hosted service that aids in establishing communications between Flash Players endpoints using RTMFP over UDP. Flash Player endpoints must stay connected to the server during the entire time of communications.

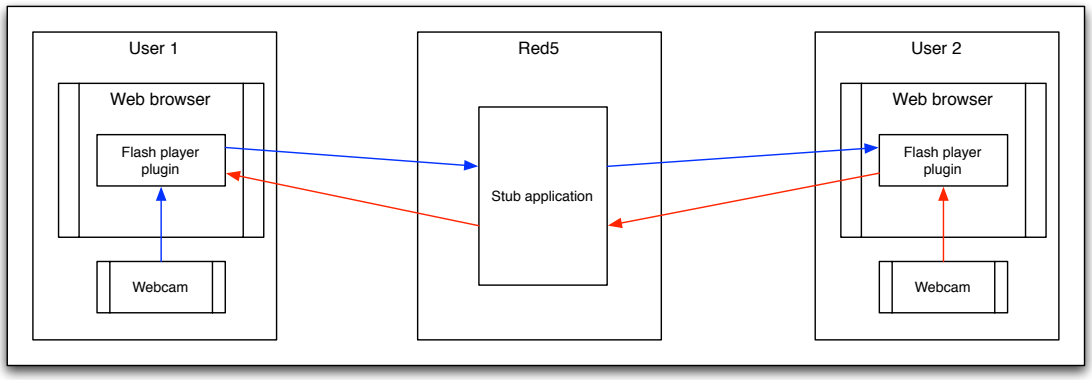


Figure 4.8: Flash RTMP videoconference architecture

Stratus is being made available as a beta service through Adobe Labs and is not yet available for distribution [str].

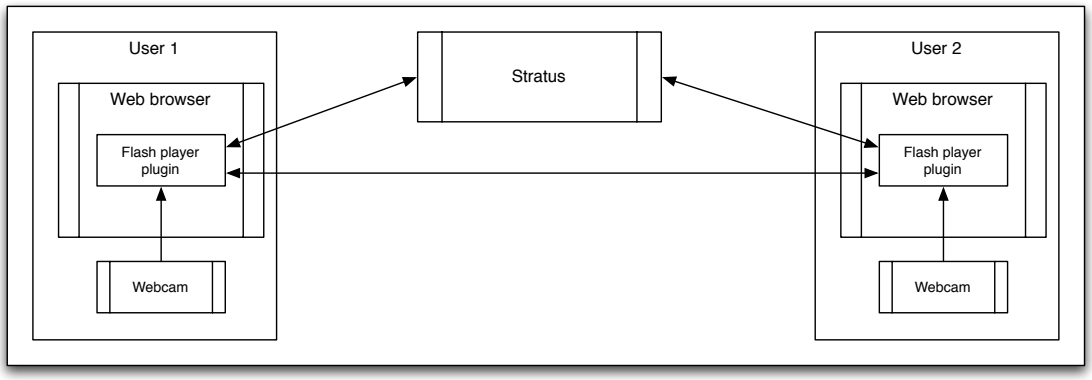


Figure 4.9: Flash RTMFP videoconference architecture

**Protocol comparison** In the context of videoconference, RTMFP has several advantages over RTMP. First, it reduces server load since all the server does is a rendezvous service and no media relay. All audiovisual media is directly sent from one client to the other, since the server helps establish a P2P connection between the two. Second, RTMFP works on top of UDP, unlike RTMP which uses TCP. UDP provides minimal latency for real-time communications. Its lesser reliability, when compared to TCP, is widely regarded as negligible in the light of the reduced latency is enables. Finally, RTMFP features data prioritization, making sure that audio is always transmitted with higher priority than video and non-critical data [Vas08].

### 4.4.2 Handling Sessions

In order for videoconference to be implemented, there must be a way to handle communication sessions. The most simple solution is to deploy a simple custom application on a remote server which works as a phone book. This system could easily be implemented using only a database and a web service. The database would contain the clients' public identifiers and information on how to reach them — a network address, stream identifier or other, depending on the videoconference implementation. The web service would receive registration and lookup queries, enabling clients to both register on the system and helping them reach their destination.

Another option is to implement session handling using SIP (Session Initiation Protocol). SIP is an application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants. These sessions include Internet telephone calls, multimedia distribution, and multimedia conferences [ea02]. If interoperability between videoconference systems is an issue, then the use of the SIP standard is recommended. This should be the option if future considerations involve several different videoconference implementations, and it would ensure the circumscribing of diverse systems which might already be deployed, especially in the case of the medical services.

### 4.4.3 Implementation considerations

Videoconference in Windows Media Center is achieved through publishing local audiovisual media to a remote streaming server, and from there streamed to the receiving endpoint. Windows Media Services, a streaming media server for Windows Server, can be used to meet this demand. It relays audiovisual media over TCP or UDP. UDP is recommended to be used for videoconference, since it privileges lower latency instead of flawless data delivery. Lower latency is usually considered to be more advantageous for the real-time communication experience, and it is the proposed setting for this purpose.

Windows Media Services has a number of advantages over other streaming media services when accessed by Windows Media clients. It allows for streaming speeds faster than the media data rate, forward error correction, and other benefits that add to the quality os service provided. It is, as such, the recommended solution for implementing videoconference when using WMC.

When using a web browser, videoconference can be enabled by using the Flash player plugin, which provides access to the locally attached audiovisual capture devices. The capture audiovisual media can be then published to a media relay server (using the RTMP protocol with a RTMP capable server such as Flash Media Server or Red5), or sent directly to the receiving endpoint using a P2P connection set up by a hosted rendezvous service (using the RTMFP protocol with Stratus, a free beta service operated by Adobe).

It is recommended to use the latter solution. It makes use of P2P, which heavily reduces the server load by delegating the media relay function to the network. The Stratus service merely serves as a rendezvous service to help establishing the P2P connection. The P2P connections run over UDP, which is deemed preferably for the reasons stated in the previous section.

## 4.5 Push Technology for the browser

Browsers don't expose socket connections to web applications for developers to exploit them using HTML, JavaScript or any other language. This has been recently challenged by the developing HTML5 standard and its WebSocket interface, introduced to enable Web applications to maintain bidirectional communications with server-side processes [Hic10]. However, HTML5 support has a long way to go before it's ubiquitous, and this is still a working draft.

In order to tackle the connectionless nature of HTTP, push technology has been developed. Push technology, or server push, describes a style of Internet-based communication where the request for a given transaction is initiated by the publisher [wikq]. There are a number of ways to implement server push. One of the most common ones is to use Comet, a web application model in which a long-held HTTP request allows a web server to push data to a browser, without the browser explicitly requesting it. Comet is an umbrella term for multiple techniques for achieving this interaction. All these methods rely on features included by default in browsers, such as JavaScript, rather than on non-default plugins [wikd].

One implementation of Comet is the Ajax Push Engine. APE is a free, open-source solution designed for Ajax Push. It includes a Comet server and a client-side JavaScript framework.

## 4.6 Discussion and conclusions

From the large number of platforms introduced on the previous chapter, only three were selected to be studied in further depth. This was due to a number of constraints and strategic decisions, presented at the beginning of this chapter. Windows Media Center, Yahoo! TV Widgets and web browsers were the chosen platforms to be thoroughly analyzed.

Windows Media Center is a good, all-inclusive solution. It allows for live broadcast TV viewing (given that a TV tuner is installed). It's a very mature software solution, which should account for stability and support. Furthermore, if endowed with the right components, videoconference support is possible.

Yahoo TV Widgets is, comparatively, a much more limited solution. Live broadcast TV viewing is possible, but developed application scopes are very limited. It is

not possible to install background applications, and these only start functioning once the user explicitly enters the widgets menu and starts it manually. Moreover, application design guidelines enforced by Yahoo make it harder to convey an application, corporate, or organization-specific image. It also places a number of constraints on user interaction models. Furthermore, this is a very recent technology, which raises questions about stability, compatibility, support and other delicate issues. Finally, there is no way to implement videoconference on this platform.

Web browsers suffer — both historically and nowadays — from a large array of compatibility issues. The effort of correctly rendering a web page the same way on all major web browsers are well known and discussed amongst web developers. Layout and rendering engines differ. Using Flash for authoring and displaying the page has been an common way to circumvent this. However, web development is relatively easy and fast to deploy, when compared to the other two platforms. Videoconference is available, even though it's a Flash implementation, which might render it unusable on browsers not supporting the Flash Player plugin.

These are then the recommended platforms for implementation of a Personal Health Channel, given present constraints. Each of them has its advantages, and selected of one over the others will eventually be done taking into consideration project-specific strategic discussion. Except for the fact that the Yahoo! platform does not support videoconference and demands more interface considerations if widgets are planned to be distributed through official channels, thorough implementation is feasible on all platforms.

# Chapter 5

# Prototype evaluation

This chapter features an evaluation of implementation prototypes developed to gain additional data for the study of the platforms. It begins by presenting the developed prototypes, and then it presents the evaluation study and results.

## 5.1 Prototypes

One of the goals of this work was to produce a prototype for each platform to be used for usability testing. These prototypes were built in the most possible agile way: development was limited, using a good-enough-for-testing approach. This was done without compromising the product's final graphical aspect or interaction models.

Yahoo! TV Widgets' prototype will not be evaluated, since it severely strays from the intended final graphical aspect and interaction model. Should this solution be seriously considered, many compromises would have to be made on both these points. As such, only the WMC and web browser platforms were used for developing and testing full prototypes.

The WMC and web browser platform prototypes implement all priority features stated on section 1.2.2, which are the following.

**View health sensor data** The application will display a chart for each of the health signs being monitored.

**See health agenda** The application will feature the user's health agenda, which is a list of next medical appointments.

**Alerts for taking medication** The application will provide users with modal alerts to remind them to take their medication.

There is a common main menu to both prototypes, which allows access to the first two features. Health sensors are grouped on a second-level menu which allows choosing which sensor to monitor. The last requirement, *Alerts for taking medication*, will generate a dismissible modal fullscreen window superimposed on the user's current view, so there is no explicit navigation into it. Backward navigation is accomplished by pressing the *back* button on the remote control. This results on the following page structure.

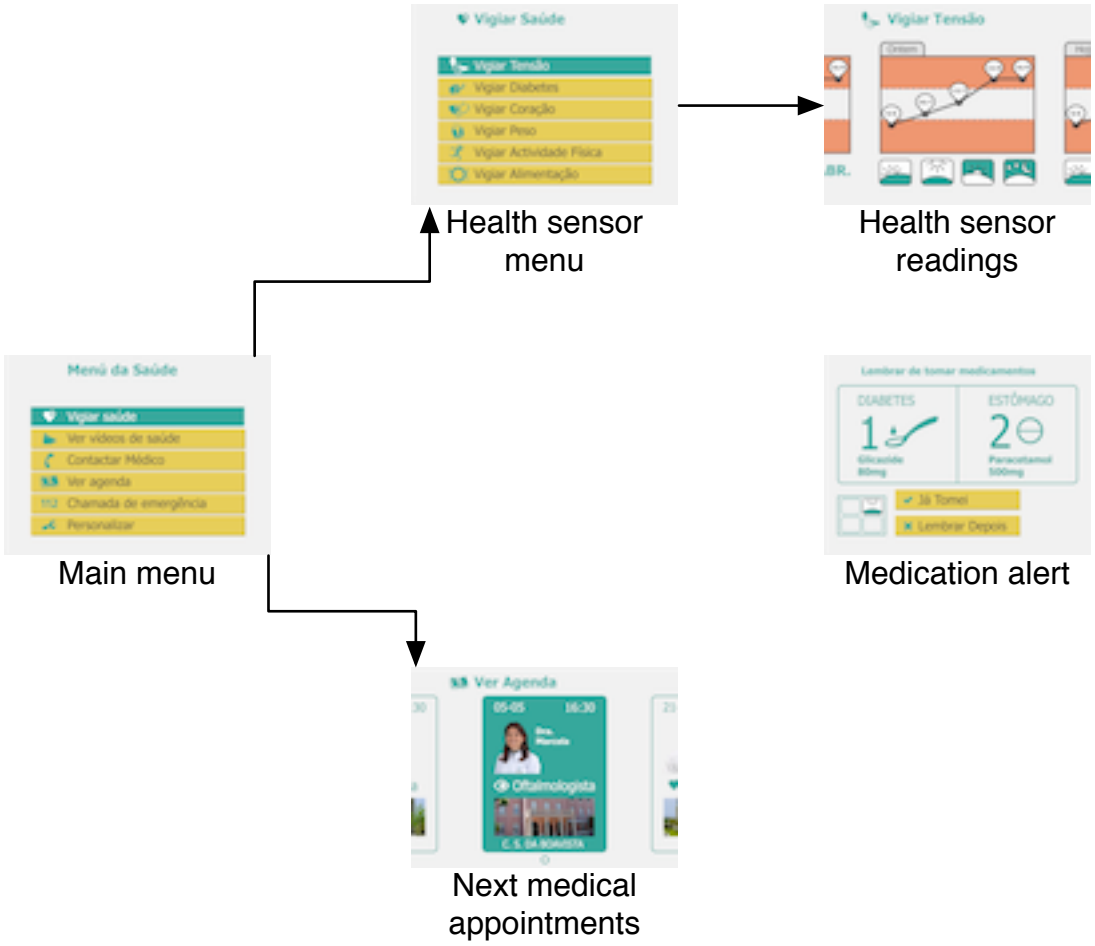


Figure 5.1: Page structure for the prototypes

The resulting prototype's graphical user interfaces are the following.


Figure 5.2: Main menu and entry point of the application.


Figure 5.3: Available sensor list.

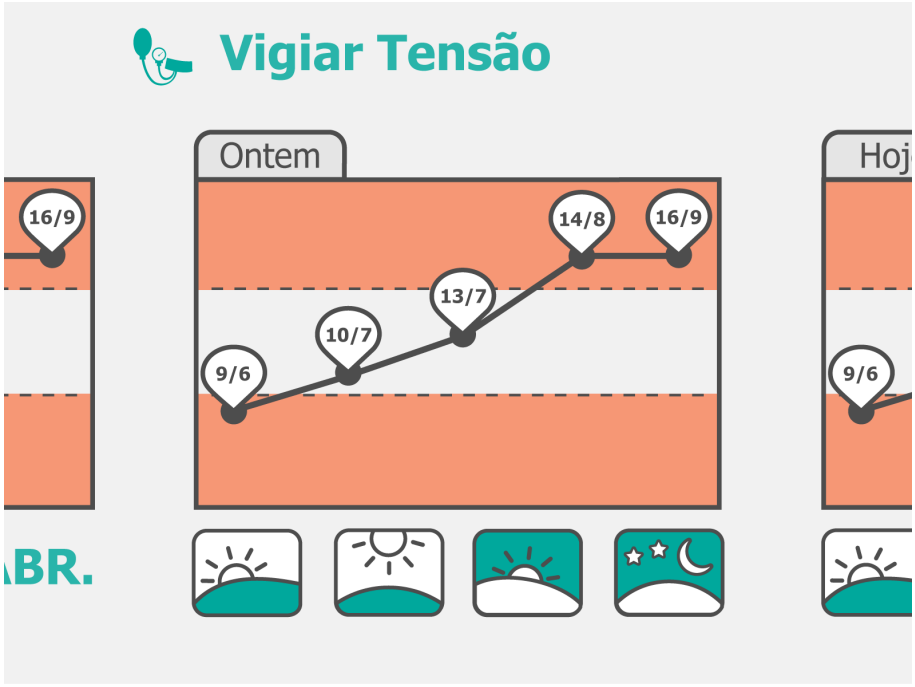


Figure 5.4: Sensor readings screen.


Figure 5.5: List of next medical appointments.


Figure 5.6: Medication alert.

## 5.2 Usability testing

Usability is a quality attribute that assesses how easy user interfaces are to use [Nie]. Jakob Nielsen defines it by five quality components. The first one, *learnability*, refers to the learning curve of the design. The more learnable a design is, the easier it is for users to accomplish basic tasks the first time they encounter such a design. *Efficiency* measures the speed of task completion once users have learned how to use the design. Another trait is *memorability*, which in its turn relates to the easiness of reestablishing proficiency in using a given design after a period of not using it. *Errors* refers to the number of errors made by users, their severity, and the way the design handles these errors. Finally, *satisfaction* evaluates the pleasantness of using the design.

It is important to focus on usability, good usability benchmarks help ensure users won't avoid the use of a product [Nie]. In the context of this project, it is important that using this system is as less cumbersome as possible. Users must feel that the usage of this system is a natural action and not a hindrance.

Evaluating the usability of a design involves *usability testing*. This is performed using a small group of representative users – whose traits resemble the design's target audience – and asking them to perform representative tasks with the design. Their actions and comments should be recorded to provide insight on the usability of the design. The test is conducted using a predefined script and on a controlled environment, to promote the usage of the same test conditions for every user.

## 5.3 Test setup

### 5.3.1 Test participants

The test participants were selected taking into account the age group target of this project's context, which is the elderly. Computer literacy is not a constant among participants, neither is STB hardware and/or software operating skills. This unbinds the obtained results from these otherwise influencing facts. The participant array is thus the following:

- Fernando Viana, 73 years old, retired. Uses the Internet for email and general browsing. Regularly operates very basic STB functions, limited to changing the channel.
- Florinda Cardoso, 68 years old, retired. Has never used the Internet or operated a STB. Florinda is unfamiliar with the operation of any electronic device.

### 5.3.2 Application access

How the users accessed the application was considered to be relevant for this study. The test simulated two modes of interaction: instant main menu access, and full platform initialization.

The first mode consists in having the participants issuing a single command to request the main menu. This was done in two ways. The first one was informing them that the product resided in a particular TV channel, and asking them to input the channel number on the remote, exactly like they would proceed to access a regular TV channel. Once they did this, the interviewer rapidly changed the television source to the laptop running a fullscreen web browser application which was connected to it, creating the illusion of a seamless transition. The other option was telling the user that a particular remote button (one disabled by default) brought up the user interface. This second option intended to simulate a remote button which brought up a STB embedded browser.

The second mode intended to reproduce the usage of WMC. A disabled remote button mimicked the *The Green Button* on the WMC remote, and the TV source was changed to the attached laptop running WMC. This brought up the WMC main menu. At this point, users were instructed of the application location and asked to start it.

### 5.3.3 Task specification

Three tasks were chosen to built the test script. These are simple tasks, but nevertheless encompass all the implemented functionalities. They are tasks expected to be performed regularly on the final system.

The first task relates to the first requirement, *View health sensor data*, and its goal is to discover a way to access the chart for a specific sensor. The chosen sensor was carefully

selected not to be the first of the list, to incite the user to actually read the menu contents and avoid random clicking.

The second task relates to the second requirement, *See health agenda*. The goal of this task is to obtain the time and place of the two next medical appointments. The rationale behind using the two next appointments and not only the next one is to incite the user to scroll through the agenda, since the immediate next appointment is the default scrolling cursor position.

Finally, the third task relates to the third requirement, *Alerts for taking medication*. This task is somewhat interwoven with the others, since it will interrupt the flow of other tasks, only to replace them with the alert modal window. This mimics the actual behavior of the final product. The implementation was slightly modified for this test, to allow triggering of an alert at the same stage for both participants. This ensures a more consistent testing process, since it exposes both users to the same conditions.

### 5.3.4 Test script

A script was developed to conduct the interviewer's procedure along the tests. This ensures that all participants have access to the same information and are treated equally. The test script follows, and it divided into parts, not necessarily sequential, for guidance. The interviewer should start with the introduction and moves on to the explanation, answer eventual questions, and move on to task direction. Further interaction may be necessary if the users is not very communicative. At the end of the test, the interviewer will use the conclusion. This task's goal is to understand wether the user understands the alert.

#### 5.3.4.1 Introduction

*Thank you for participating in this study. Its objective is to evaluate a product which allows users to manage their health using their television. The results of this study will contribute to the improvement of such a product, making sure that the final product is as easy to use as possible.*

*I am going to ask you to operate this remote control as if it were the one from your television.* [Presents user with remote control. Further explanation of the remote might be offered if the user proves to be unfamiliar with the concept of directional keys.] *I am then going to give you some simple tasks to see how the system responds. Please make sure that, during this procedure, you think out loud, verbalizing everything you might be thinking whether you think it might be relevant or not. Everything you say can be very important for improving the system. I will be registering your actions and comments.*

#### 5.3.4.2 Explanation

*Let's begin with a small presentation of this product. It is actually a study that will lead to a system that helps people to manage their health using their television. It allows people to see their health agenda, engage in videoconference sessions with their doctors, receive alerts to take their medication, among many other helpful features.*

*This study is important since it will help us to determine if we are going the right way with this product. Since it is a product directed to people, it is important that we make sure that people can easily and pleasurably use it.*

*In a moment, I will give you some instructions and ask you to follow them. You should use only the remote control I gave you to complete these tasks. I want to assure you that, in this study, it is the product that's being tested, not you. If you are faced with any difficulty or can't conclude a task, it's the product's fault, not yours. The product can only be seen as successful if people can use it without any problems.*

*Please face each task at your natural pace. There is no need to hurry. Make yourself comfortable, as if you were watching television. However, if you exceed five minutes to perform a task, I will ask you to move on to the next one. If, at any time, a task makes you uncomfortable, or you feel frustrated and don't want to go on, please say so and we will move on to the next one.*

*Before you go on, do you have any questions?*

#### 5.3.4.3 Exploration

*Before we start with the tasks, I'd like you to explore the product at your will, using only the remote control, for a couple of minutes. Please think out loud, telling me what you think of the product and its features. Please be aware that many of the product's features aren't yet built, and may be impossible to use. When you feel you've explore enough, please say so and we will move on to the tasks.*

#### 5.3.4.4 Application access

*This part of the procedure will help us understand how we should make the system start. For now, return to the original remote from your television. The system is available on channel 112. Please, insert that number on your remote in order to access it.* [Switches the TV source as soon as the user enters 333 on the remote.] *Thank you. Still on the same remote, this button* [identifies button] *will also start the system. Please, press it.* [Switches to the TV source as soon as the user presses the button.]

*Very good. Now, we are going to test yet another way to start the system. This button* [identifies button] *will bring up another slightly different system. Please press it.* [Switches to the TV source as soon as the user presses the button.] *Now, I am going to*

*ask you to use the remote I gave you to access the system this time. It is located up there inside the EXTRAS menu, and its button has a white lighting bolt on a black background.*

#### 5.3.4.5 Task direction

*Feel free to ask any question you feel like about any task. Upon completing the task, please return to the same point you started on.*

**First task** *Ok, first task. Please tell me how your heart was doing yesterday.*

**Second task** *Now let's move on to the second task. Please tell me when are your next two medical appointments.*

**Third task** [Just after the conclusion of the second task, in order to surprise the user.] *Oh, I had a surprise in stock for you. Do you understand what just happened?* [Register statement.] *Please ask the system to remind you later.*

#### 5.3.4.6 Further interaction

If the participant isn't being very responsive, the interviewer may incite him or her to talk using sentences like *What are you thinking?* or *Do you feel stuck?*, depending on the context.

#### 5.3.4.7 Conclusion

*Once again, I'd like to thank you for having participated on this test and helping us improving this product. Your contribute has been extremely valuable to us and will certainly be taken into consideration for future development and perfecting this product.*

## 5.4 Test procedure

This usability test occurred under controlled conditions. External influences were minimized to a large extent, asking the participants to switch off their cellphones and avoiding interruptions.

The participants were guided by the interviewer using the script provided on the previous section. They were asked to think out loud, stating what was on their minds during the procedure of the test.

The interviewer registered the participants' performance of the tasks, noting wether the users completed the tasks and how long did it take for said completion. The interviewer also kept track of relevant participants' comments and difficulties met.

## 5.5 Test results

### 5.5.1 Application access

Inputting the channel number or pressing a button for immediate main menu access was accomplished without any complications. However, navigating through WMC proved to be a slower method for both participants, and Florinda required additional guidance to be able to access the application.

### 5.5.2 Navigation

Florinda was unable to understand navigation metaphors that are otherwise clear to computer users. Firstly, she repeatedly mistaken the first highlighted item of a menu as being part of the title of the page instead of a button, even after observing how the highlight shifted to other items when the directional keys were pressed. She made this assumption repeatedly through the application. She eventually got hold of the directional interaction model, but only when given several hints towards its operation. Another issue she showed trouble with was the concept of activating a menu item after having set the cursor on top of it. Several times she had to be hinted that she was almost there, and just had to press the button, and would sometimes press the up or down keys and hence going back a step by removing the cursor from the correct item. She did not have any problems with navigating to previous views.

Both users manifested a certain hesitation in navigating back to the main menu from the sensor menu.

### 5.5.3 Task execution

Both users had to be encouraged to use the left and right directional keys to navigate through the lists of both health sensor readings and medical appointments. Otherwise, Fernando accomplished all tasks without effort. He clearly understood where each menu item was supposed to lead, and comfortably navigated through the application to obtain the necessary answers. Florinda, however, required constant support to be able to apprehend the concept of navigation, as she constantly demonstrated constant problems with navigation metaphors. Florinda did, eventually, accomplish all tasks, even if constantly assisted on navigation.

### 5.5.4 Relevant participant comments

Both participants clearly preferred to access the application using either a channel number input or pressing a single button. They disliked having to navigate through WMC to

access the application they wanted to use. They also mentioned that they couldn't tell apart the main menu from the sensor menu very well due to their high likeliness.

Fernando mentioned that his good performance was due to his familiarity with computers, even if he's only a basic user. He stated that wouldn't otherwise be able to operate the system.

Florinda said that she would have been unable to complete the tasks if the interviewer hasn't offered so much guidance on navigating. She said that she had no problem understanding what the labels meant, but didn't understand very well how to jump from one place to the other.

## 5.6 Result discussion and conclusions

Even though the tasks were successfully accomplished, the fact that the interviewer had to provide hints suggests that there is still room for improvement. The performed evaluation allows for the drawing of several conclusions.

Navigating through the WMC interface to access the application was perceived as cumbersome and unnecessary. However, this might have been because the participants were unaware of its extra features.

The fact that both users needed to be hinted to used the left and right directional keys to navigate through the lists of both health sensor readings and medical appointments shows that the interface hints (the partially shown list items on the edges of the screen) are not enough, and should be improved. Visual arrows might encourage usage of the directional keys.

Florinda's navigation difficulties provide the most useful insight. It shows that computer illiteracy can lead to a steep learning curve of standard operation modes. Assuming that unfamiliarity with even the most simple of digital graphical interfaces is common among the elderly, this severely challenges the evaluated interface. These navigation hurdles could be overcome in a number of ways. The first is training. This is a viable option but a costly one, both in time and resources. The interface navigational paradigm could be improved in order to shift to an easier to use one. These are several ways this could be achieved, namely through the following.

**Using a touchscreen interface** This might simplify interaction with the application by providing a more intuitive way to select menu items, since the user just had to touch the feature he or she intended to use. However, it is not the best solution, since the application is designed to use on a television.

**Numbering the menu items** Menu items could be labeled with numbers, which the user would select using the regular numbered buttons on the remote control. A way to prevent this from changing the channel would, however, need to be considered.

**Using voice commands** The application could be altered to recognize voice commands emitted by the user. This way, the application would handle navigation by itself.

Besides the aforementioned comments, the graphical interface could also be improved in general by reserving an area of the screen to indicate the possible actions that can be performed on a certain page. On menus, up and down arrows might hint users of how the menu can be navigated, as well as help them recall the interaction mode. Lists could feature the same arrows, but pointing left and right. An indication that buttons are clickable is not as clear to implement, though. A workaround for this could be the implementation of a timer which would automatically select the menu item the cursor was on after a certain period of time, but this might confuse users who are expecting the application to respond only when they explicitly command it to.

Another possibility is the activation of a context-aware help button. The purpose of this feature would be to replace the interviewer in hinting the users, either displaying on screen or speaking out loud interface hints to guide the users through the application. For these prototypes, these texts' semantic equivalents would be the following.

**Main menu context** *Use the up and down buttons* [possibly depicting them on screen at this time] *to make the function you want to use turn green, and then press the OK button* [possibly depicting it on screen at this time] *to use it*.

**Sensor menu context** *Use the up and down buttons* [possibly depicting them on screen at this time] *to make the sensor you want to watch turn green, and then press the OK button* [possibly depicting it on screen at this time] *to use it*.

**Sensor readings context** *Use the left and right buttons* [possibly depicting them on screen at this time] *to watch this sensor's readings on different periods of time*.

**Health agenda context** *Use the left and right buttons* [possibly depicting them on screen at this time] *to see other appointments you have*.

# Chapter 6

# Conclusions and future work

This chapter presents the conclusions of this work. It also provides considerations on where there is room for improvement.

## 6.1 Conclusions

The present work has seen its goals fulfilled. The chosen platforms from a survey of commercial software and hardware platform offers were analyzed thoroughly in order to provide a framework for the Personal Health Channel implementation, respecting its parent project's constraints.

From all the COTS solutions available, only some were, at the same time, available for further research and a sensible match for the project — Windows Media Center, Yahoo! TV Widgets, and web browsers. Some apparently adequate platforms, like Microsoft Mediaroom, play a small part on this work due to availability or legal issues.

These platforms' technological backgrounds were comprehensively studied in order to enable future development of the planned Personal Health Channel features. Most features aren't very demanding and its deployment was found to be possible in all platforms. Videoconference, being a technologically heavy feature, was given special consideration, and found to be possible to implement in all platforms except for YW.

Prototypes was built for the priority features, and subject to usability testing. The insight they provided was an important contribution not only for the design of the user interface of the Personal Health Channel, but also for the underlying platforms. Users tended to prefer the more seamless integrations of the system on their TV.

This work proves that it is possible to develop and deploy the Personal Health Channel resorting to a number of COTS solution, avoiding more costly and development-heavy alternative solutions such as developing a system from scratch. It also shows that the available COTS solutions are flexible enough to allow for all the proposed features to be implemented, and provides a framework for this implementation — from proposing an

architecture for data feeding to meticulously studying the selected platforms. Finally, this work presents thorough considerations on the best videoconference deployment solutions interoperable with these platforms.

## 6.2 Future work

Future developments will be the actual implementation of the Personal Health Channel on the context of the eCAALYX project, and will follow immediately suit. The development considerations exposed on this document will streamline this future development, ensuring it is done on top of the most adequate platforms in time for the field trials next year.

If project deadlines are extended, other — currently unavailable, yet promising — platforms may be studied, such as Microsoft Mediaroom. These platforms might prove to be more aligned with the project's goals that the solutions proposed along this work.

# References


[ado] Opentv and adobe collaborate to advance rich television experiences. http://www.microsoft.com/Mediaroom/.

[Bou09] Magel N. Kamel Boulos. An enhanced ambient assisted living experiment for older people with multiple chronic conditions. Technical report, eCAALYX Project Consortium, 2009.

[boxa] Boxee box. http://www.boxee.tv/box.

[boxb] Boxee devwiki. http://developer.boxee.tv.

[boxc] Boxee faq. http://www.boxee.tv/faq.

[Bro96] Nancy Brown. Telemedicine 101. Updated on January 13, 2005, 1996.

[Car99] A Carmichael. Style guide for the design of interactive television services for elderly viewers. http://www.computing.dundee.ac.uk/projects/UTOPIA/publications/Carmichael%20-%20DesignStyleGuideFinal.pdf, 1999.

[cea] Cea announces new home networking standard, remote user interface for the digital home. http://www.ce.org/PDF/CEA-2014_Press_Release_Detail.pdf.

[CEA07] CEA. Web-based protocol and framework for remote user interface on upnptm networks and the internet (web4ce). Technical report, CEA, 2007.

[Cho] Dennis Chominsky. Designing graphics for video. Published by Prentice Hall Professional.

[Don80] Avedis Donabedian. *The Definition of Quality and Approaches to its Assessment*. Ann Arbor: Health Administration Press, 1980.

[dota] Assemblies in the common language runtime. http://msdn.microsoft.com/en-us/library/hk5f40ct(VS.80).aspx.

[dotb] Assembly benefits. http://msdn.microsoft.com/en-us/library/6h38y9z9(VS.80).aspx.

[ea02] Rosenberg et al. Sip: Session initiation protocol. Technical report, Internet Engineering Task Force, 2002.

[eca] About the project. http://ecaalyx.org/index.php?option=com_content&view=article&id=44:article2&catid=34:public-documents&Itemid=34.

[goo] Google tv faq. http://www.google.com/tv/faq/.

[Gro09a] Intel Digital Health Group. Intel health care management suite. Technical report, Intel, 2009.

[Gro09b] Intel Digital Health Group. Intel health guide phs6000. Technical report, Intel, 2009.

[Han06] Vibeke Hansen. Designing for interactive television. Technical report, BBC, 2006.

[Hic10] Ian Hickson. The websocket api. Technical report, World Wide Web Consortium, 2010.

[HJ88] Doran MC. Hall JA. Meta-analysis of satisfaction with medical care: description of research design and analysis of overall satisfaction levels. *Social Science and Medicine*, 1988.

[itv] Set-top boxes. http://www.itvdictionary.com/set-top_box.html.

[jav] Javafx faq. http://javafx.com/faq/.

[KBLA+09] Maged N. Kamel Boulos, Ricardo Castellot Lou, Athanasios Anastasiou, Chris D. Nugent, Jan Alexandersson, Gottfried Zimmermann, Ulises Cortes, and Roberto Casas. Connectivity for healthcare and well-being management: Examples from six european projects. *International Journal of Environmental Research and Public Health*, 6(7):1947–1971, 2009.

[LGF+08] Hyowon Lee, Cathal Gurrin, Paul Ferguson, Sorin Sav, Thomas Foures, Stephane Lacote, Noel E. O'Connor, Alan F. Smeaton, and Heeseon Park. Balancing simplicity and functionality in designing user-interface for an interactive tv. In *6th European Interactive TV Conference (EuroITV 2008)*, Salzburg, Austria, 3-4 July 2008 2008. Tampere International Center for Signal Processing, Tampere International Center for Signal Processing.

[LJM+05] Karyn Y. Lu, Dr. Janet, H. Murray, Dr. Peter Mcguire, and Allison Dollar. Interaction design principles for interactive television approved by:, 2005.

[LLCD07] George Lekakos, George Lekakos, Konstantinos Chorianopoulos, and Georgios Doukidis. *Interactive Digital Television: Technologies and Applications*. IGI Publishing, Hershey, PA, USA, 2007.

[med] Microsoft mediaroom. http://www.microsoft.com/Mediaroom/.

[Mil01] Edward Alan Miller. Telemedicine and doctor–patient communication: an analytical survey of the literature. *Journal of Telemedicine and Telecare*, 2001.

[myka] Myka developer program. http://www.myka.tv/developers.html.

[mykb] Myka ion. http://www.myka.tv/myka-ion.html.

[myt] Mythtv, open source dvr. http://www.mythtv.org/.

[neu] Neuros link. http://www.neurostechnology.com/.

[Nie] Jakob Nielsen. Usability 101: Introduction to usability. http://www.useit.com/alertbox/20030825.html.

[opea] Opentv. http://www.opentv.com/.

[opeb] Opentv community portal. http://community.opentv.com/.

[opec] Opera widgets sdk - develop one app for one web. http://www.opera.com/media/b2b/Opera-Widgets-SDK-product-sheet.pdf.

[oped] Opera widgets technology. http://www.opera.com/business/solutions/widgets/technology/.

[PBBL89] Roxanne Parrott, Judee K. Burgoon, Michael Burgoon, and Beth A. LePoire. Privacy between physicians and patients: More than a matter of confidentiality. *Social Science & Medicine*, 29(12):1381 – 1385, 1989.

[Per95] D.A. Perednia. Telemedicine technology and clinical applications. *Journal of the American Medical Association*, 1995.

[phi] Net tv specifics. http://www.consumer.philips.com/c/about-philips-nettv-partnerships/32374/cat/pt/.

[pop] Popbox. http://www.popbox.com/.

[PS95] Dena S. Puskin and Jay H. Sanders. Telemedicine infrastructure development. *J. Med. Syst.*, 19(2):125–129, 1995.

[RL.95] Bashshur RL. On the definition and evaluation of telemedicine. *Telemedicine Journal*, 1995.

[str] Stratus:faq. http://labs.adobe.com/wiki/index.php/Stratus:FAQ.

[Tra03] V Traver. Multiagent home telecare platform for patients with cardiac diseases. *Computers in Cardiology*, 2003.

[tvc] Television broadcast stations. https://www.cia.gov/library/publications/the-world-factbook/fields/2015.html.

[tvo] Multichannel video programming distributor. http://en.wikipedia.org/wiki/Multichannel_video_programming_distributor.

[tvp] Access to information: Television sets per 1000 people. http://earthtrends.wri.org/text/population-health/variable-558.html.

[Unk] Unknown. What is telemedicine? Published by ICUcare LLC.

[Vas08] Jozsef Vass. Stratus service for developing end-to-end applications using rtmfp in flash player 10. Updated on 26 April 2010, 2008.

[wika] Ajax (programming). http://en.wikipedia.org/wiki/AJAX.

[wikb] Boxee. http://en.wikipedia.org/wiki/Boxee.

[wikc] Ce-html. http://en.wikipedia.org/wiki/CE-HTML.

[wikd] Comet (programming). http://en.wikipedia.org/wiki/Comet_(programming).

[wike] Comparison of web browsers — javascript support. http://en.wikipedia.org/wiki/Comparison_of_web_browsers#JavaScript_support.

[wikf] Google tv. http://en.wikipedia.org/wiki/Google_TV.

[wikg] Interactive television. http://en.wikipedia.org/wiki/Interactive_television.

[wikh] Interlace. http://en.wikipedia.org/wiki/Interlace.

[wiki] Managed code. http://en.wikipedia.org/wiki/Managed_code.

[wikj] Microsoft mediaroom. http://en.wikipedia.org/wiki/Microsoft_Mediaroom.

[wikk] Mythtv. http://en.wikipedia.org/wiki/MythTV.

[wikl] Neuros link. http://en.wikipedia.org/wiki/Neuros_Technology#Neuros_LINK.

[wikm] Ntsc. http://en.wikipedia.org/wiki/NTSC.

[wikn] Opentv. http://en.wikipedia.org/wiki/OpenTV.

[wiko] Pal. http://en.wikipedia.org/wiki/PAL.

[wikp] Persistence of vision. http://en.wikipedia.org/wiki/Persistence_of_vision.

[wikq] Push technology. http://en.wikipedia.org/wiki/Push_technology.

[wikr] Real time messaging protocol. http://en.wikipedia.org/wiki/Real_Time_Messaging_Protocol.

[wiks] Red5. http://en.wikipedia.org/wiki/Red5.

[wikt] Set-top box. http://en.wikipedia.org/wiki/Set-top_box.

[wiku] Videoconferencing. http://en.wikipedia.org/wiki/Videoconferencing.

[wikv] Web browser. http://en.wikipedia.org/wiki/Web_browser.

[wikw] Web browser engine. http://en.wikipedia.org/wiki/Web_browser_engine.

[wikx] Windows media center. http://en.wikipedia.org/wiki/Windows_Media_Center.

[wiky] Windows media encoder. http://en.wikipedia.org/wiki/Windows_Media_Encoder.

[wikz] Xbmc. http://en.wikipedia.org/wiki/XBMC.

[wmca] Application types. http://msdn.microsoft.com/en-us/library/ee525789.aspx.

[wmcb] Overview of the windows media center platform. http://msdn.microsoft.com/en-us/library/ee526021.aspx.

[wmcc] Use a remote control with windows media center. http://windows.microsoft.com/en-us/windows-vista/Use-a-remote-control-with-Windows-Media-Center.

[wmcd] Windows media center. http://www.microsoft.com/windows/windows-media-center/.

[wmce] Windows media center sdk overview. http://msdn.microsoft.com/en-us/library/ms816327.aspx.

[WR97] Darkins A. Wootton R. Telemedicine and the doctor–patient relationship. *Journal of the Royal College of Physicians of London*, 1997.

[xbm] Xbmc. http://xbmc.org/.

[yw:a] Connected tv business opportunity. http://developer.yahoo.com/connectedtv/business/index.html.

[yw:b] Connected tv forum. http://developer.yahoo.net/forum/?showforum=91&endsession=1.

[yw:c] Device interface. http://developer.yahoo.com/connectedtv/devguide/CTV_DG_Device_Interface.html.

[yw:d] Tv widgets overview. http://developer.yahoo.com/connectedtv/devguide/CTV_DG_Overview.html.

[yw:e] Yahoo! connected tv developer guide. http://developer.yahoo.com/connectedtv/devguide/YWE_TV_Widget_Developer.pdf.

[yw:f] Yahoo!(r) brings the cinematic internet(tm)to lifeand revolutionizes internet-connected television. http://yhoo.client.shareholder.com/press/releasedetail.cfm?ReleaseID=358066.

# REFERENCES

# Appendix A

# Test bed setups

As discussed on chapter 1, field trials for this solution will be performed in a near future. Test beds must thus be conceived to enable the trials to succeed. This section presents a setup proposal for these test beds.

The back-end system architecture proposal presented on the previous section is assumed to remain the same whichever platform is selected, so no further specification will be necessary. However, the test beds will differ with the selected platform. Hence, a different setup will be presented for each one.

Both the medical services and the users's premises must be equipped with an Internet connection in order to enable the following setups.

## A.1 Windows Media Center

Windows Media Center requires an underlying Windows installation to run. It is thus imperative that each user has a PC attached to a TV set. This PC need not boast high-end hardware, since its only purpose will be to serve as a container for WMC, and should only comply with the necessary requirements stated by Microsoft in order to keep costs to a minimum. The same case could be made for the Windows versions, which should be the least expensive one: in the case of Windows 7, the latest release, this is Windows 7 Home Premium.

To interact with WMC, the setup should also include a WMC remote control. These remotes feature a special button called *The Green Button*, which pressing is required to start WMC, and provide a way for the user to interact with it.

## A.2 Yahoo! TV Widgets

This platforms requires the acquisition of compatible TV sets. It is recommended that all users have the same model, to minimize compatibility issues. The TV sets come with a remote control to interact with the widgets, so no additional hardware is needed.

## A.3 Web browser

The Web browser is the platform that offers more configuration possibilities. The browser can either be installed on a PC or a STB. In order to make the user experience when

interacting with the personal health channel as seamless as possible, it is recommended to use a STB with an integrated browser. The cost will also be lower than what the use of a PC would involve, and remote control interaction will be integrated.

Should the browser be installed on a PC, a remote control will have to be purchased separately to enable the user to interact with the Personal Health Channel remotely.

There is also the possibility to use a separate computer, together with a separate display, as an information appliance dedicated to the Personal Health Channel. This would require a keyboard attached for interaction. A touch screen display would provide an easier interaction, but increase the cost of deployment.

# Appendix B

# Television interaction design guidelines

The particular way users interact with TVs justifies devising a set of guidelines to help designers and developers to build graphical and physical interaction models for the TV. This section provides an overview of some of the most important guidelines. Only platforms which support the implementation of these guidelines should be considered for deployment.

## B.1 General considerations

### B.1.1 Resolution

TV screens have historically been built with much lower resolution than PC screens. The lingering of analog TV contributed to the maintenance of these low standards, since even PAL[1], whose resolution is higher than NTSC's[2], is limited to 576 horizontal lines. Only in some countries higher definition television was broadcast analogically. Most countries undergo a shift towards digital television before enhancing transmission resolutions. Unfortunately, this doesn't guarantee TV set upgrades, which means that we must account for low resolution display still being present in a lot of households.

Another issue is the pixel aspect ratio. Computer monitors use pixels that are square; on a television screen they are slightly rectangular, roughly 1.067 times as wide as they are tall. To get around this disparity, all images destined for television but initially created on a computer should be adapted [Han06].

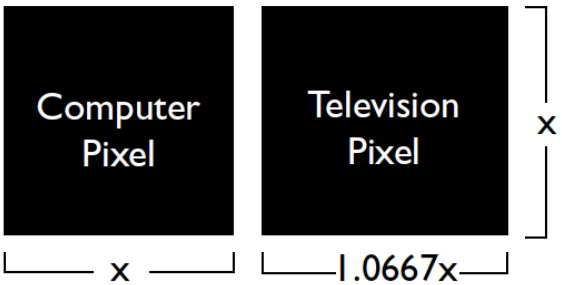


Figure B.1: Computer and television pixels.

[1] PAL is an analogue television encoding system used in most of Europe[wiko].
[2] NTSC is an analogue television encoding system used in most of the Americas[wikm].

### B.1.2 Security margins

Most televisions, especially older models, cut off the outer edges of the video screen [Cho]. Security margins should thus be taken into consideration so as to make sure that critical content is always displayed. [Cho] recommends accounting for at least a 15% border around all four sides of the outer video edge. Important content items should never be placed outside this margin.

### B.1.3 Aspect ratio

Although an increasing number of the TV viewing population currently owns a widescreen television [Han06], there are two main considerations to ensure graphics display correctly when viewing widescreen content on a 4:3 television. First, the left and right edges of the screen will not be visible on any set where the image is cropped to 4:3 (Centre Cut-Out). These areas should be treated as additional safe margins and should only contain background information. Text, color keys, navigation and all other essential graphical elements must be kept in the 4:3 safe area. Second, the entire application may be shrunk 25% to fit within a letterbox format on a 4:3 set. In this case, any text in the video must be large enough to stay legible at a smaller size. Therefore the font size used should be no less than 27 pt when designing for widescreen video streams, to allow for this potential size reduction [Han06].

### B.1.4 Flickering

Analog television broadcasts, as well as bandwidth-saving digital television, transmit interlaced video. Interlacing is a technique of improving the picture quality of a video signal without consuming extra bandwidth [wikh]. It consists of dividing each frame in two *fields*, with half the vertical resolution each, and displaying them one after another. This way, each frame is actually only half of a full image. However, the persistence of vision phenomenon, which states that an afterimage is thought to persist for approximately one twenty-fifth of a second on the retina, creates an illusion of continuity [wikp].

Because of interlacing, single pixels will flicker. To reduce interlace flicker, designs for TV-safe graphics should be no less than 2 or 3 pixels wide. Also, it is best to avoid detail, since images with fine details will blur and television viewers won't be able to see them properly [Han06, LJM+05].

Due to the flickering of single pixels, applying dither to images should be avoided. The designer should refrain from using intricate patterns on screen, as this will cause a Moiré distortion. This effect is a common problem and occurs when regular patterns such as grids or dots are rotated away from the true vertical [Han06].

### B.1.5 Bloom

In the case of analog TV, each scan line is made up of an analogue signal, which controls changes in color and value across the screen. Strong contrasts in hue or luminance along these lines can cause distortion, throwing the display of vertical edges out of alignment. The resulting bloom causes curves or waves to appear in vertical lines. To avoid this problem, designers should avoid making strong changes in color along vertical edges. Text in strong colors near rectangular edges can cause especially bad distortion [Han06].

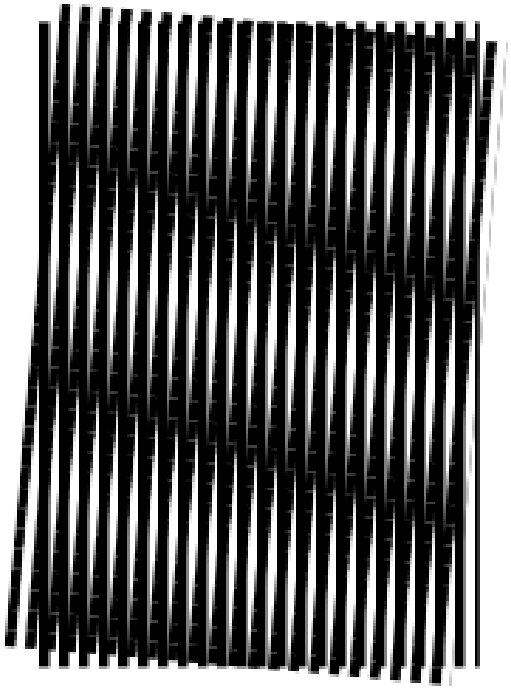

Figure B.2: The Moiré effect.

### B.1.6 Color

Use of color has to be considered carefully. Television screens have a more limited overall gamut than computer monitors and a much higher gamma value. This results in much higher contrast and saturation levels during display. To achieve parity in terms of color intensity, images should be toned down and desaturated when taken from the computer to the television screen. Vivid reds and oranges cause particularly bad distortion and pure white and black should always be avoided. The strongest white used for television display should hold a value of around 95%, or 240/240/240 in RGB terms. The darkest black conventionally used should hold a value of 5%, or 16/16/16 in RGB terms [Han06].

### B.1.7 Typography

Text poses difficult challenges on television screens, as viewers are not accustomed to reading static blocks of text on screen. Other contributing factors include poor display quality of still images, the difficulty of rendering detail at low screen resolutions and interlacing flicker. As a general rule, fonts must be used with care. Very light weights or fonts with very narrow and broad strokes should be avoided. It is a good idea to use simply constructed sans-serif fonts and use anti-aliasing to increase readability. It is thus considered good practice to use the following guidelines [Han06, LJM+05]:

- body text should not generally be smaller than 24 point (this is not consensual: other authors suggest an 18pt minimum [LGF+08]);

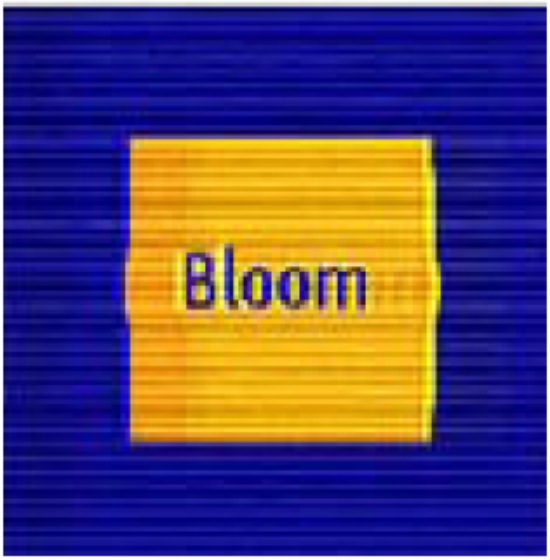


Figure B.3: The bloom effect.

- no text should ever be smaller than 18 point under any circumstances;
- light text on a dark background is slightly easier to read on screen;
- text on screen needs looser leading (greater line spacing) than in print;
- when technically possible, tracking should be increased by up to 30%;
- a full screen of text should contain a rough maximum of 90 words;
- text should be broken into small chunks that can be read almost instantly.

## B.2 Interaction

Interaction with a TV is usually accomplished through the use of a remote control. Its input speed is much slower than that of a keyboard or a mouse, and a number of considerations arise from this fact. Special care should be taken in respect to the attention span and focus of TV users when designing TV applications or services. Good practices include:

- using a shallow or flat menu hierarchy [LGF+08];
- designing the service so that it is operated only with the keys that can be found in each remote control type, and using non-standard keys only to provide additional functionality like shortcuts [LLCD07];
- enabling the user to choose when to watch the content, not using predetermined timing for presenting the material, and making sure the users have enough time to interact with the service [LLCD07];
- making input easy, preferring word selection over text input and giving instructions on the expected input format like *DDMMYY* [LLCD07];

- making it easy for the user to correct input errors [LLCD07];
- displaying the most important instructions in the user interface: all information on screen and even empty spaces around objects serve as a clue to how to use the service [LLCD07];
- presenting a single message on a single screen [Car99];
- enhancing the perception of visual highlights by using sound [Car99].