Product Vision and Planning

Willem Vaandrager 4175115 Elgar de Groot 4091108
Johnny Verhoeff 4137175 Hugo Reinbergen 4161173
Koos van der Linden 4133145

March 13, 2013
first draft

Abstract

Contents

Chapter 1

Introduction

A good software product requires not only good code, it needs to run on the right hardware and that hardware can be spread among different platforms. This needs to be managed in a way that is maintainable and cost-effective. The data needs to be stored in databases and those need to be managed as well.

This document will describe the implementations of such systems for this project. It will start with evaluating the various sub-systems and how these are dependant of each other. Then, the way those sub-systems are mapped to computers, processes and how the communication between multiple systems is handled is covered. Following the database will be described. The design of the data and how this is stored on big and small scale. The types of files used by the systems will also be specified. As last item, the concurrency between processes

1.1 Design goals

The goals behind designing these systems is to get a clear idea what is required to be made to get the eventual product to function properly. It has to be clear how much time will have to be spent on making systems such as databases and sub-systems so that the creation and implementation can be spread equally among the sprints. This document offers a form of assurance where there will be less surprises during the development process when it comes to accepting certain data structures and systems. This way, problems where finished work ends up being a waste of time due to changing systems half way through will occur far less likely and in fact, most likely not at all. Finally, this architectural design presents another opportunity to show our plans for the development process to the product owner to create more transparency and options for communicating about the project.

Chapter 2

Software architecture views

2.1 Subsystem decomposition

In order to build the eCoach program, this tasks has to be divided in smaller sub problems. So the system is divided into subsystems. The choice has been made to design he system by use of a model-view-controller-model (MVC). One of the reasons for this approach is the need of two different representations of the data for the therapist and the patient: two different views on the same data.

So for the view-layer at least two different views will be created, one for the therapist and one for the patient. The patient view-layer will use the avatar and in this project the avatars display is realized by remote procedure calls to an engine called vizard. To ease the replacement of this engine, communication from the view subsystem with this system has to be done through an interface. This interface is a sub system of the view layer.

The controller-layer can be divided into the different tasks the system has to accomplish. A communication system to transfer data from the patient to the therapist and back is one of the subsystems. Another subsystem is the behaviour logic of the coach avatar. The eCoach has to behave as a real coach would do.

The model-layer consists of the database system to store all the information. For now, the model-layer is not sub divided. In section 2.3 a further description of this layer is given.

2.2 Hardware/software mapping

2.3 Persistent data mapping

For this project we will have to use a database of some sort, because there is a lot of data that has to stored en accessed later on. This data may also have to be accessed at an other location for example, the patients progress has to be viewed by the therapist and the therapist needn't be at the same location. The team's preference for a database type is a SQL database that is "in the cloud". We have chosen for this type because we all have experience with SQL and is relatively easy to understand. We have chosen for "in the cloud" because than it will be accessible everywhere and we have no maintenance for the server and we don't even have to buy a server or domain. The login names and passwords for each patient is stored in the database, so the patient can login wherever they are (if they have the software of course). For each patient his/her progress is also stored in the database, this is very important for the therapist because they don't see each other on a regular basis. The social events outside of the therapy will also be saved, so the avatar can 'remember' it and talk about it with the patient some time in the future.

Because the server or the patients internet connection can be offline, the patients progress and everything else that can be stored in the database, is first saved locally at the patients hard drive. And than it is synced with the server.

2.4 Concurrency