# 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 paragraph 2.3 a further description of this layer is given.

# Hardware/software mapping

The different layers explained in the previous session are the divisions that can be made *inside* the code. But how is the system organized in term of processes and hardware?

For instance for hardware there will be at least three different computers/types involved:

* The computer of the therapist, used to display the progress of all patients and to communicate with them
* Numerous computers of patients, used to work with the eCoach and to communicate with the therapist
* Some kind of server used to facilitate the communication

On the computer of the therapist and on the computer of a patient a database system will run to store all the permanent user data (more about this in paragraph 2.3). No decision has yet been made about what kind of database system is going to be used.  
On the patient system a vizard process will run to display the avatars. Most of the view-layer (without the avatar displaying) and the control-layer is run from the same main process.   
Because the patient and the therapist do not share a computer and because the use of the program is very different for the patient and the therapist, separate programs are made for both users. This decreases the chance of security leaks and also decreases the amount of files to be installed localy.

Glossary: Model-view-controller-model: A design pattern which separates the data (model) from the representation (view) and the programlogic (controller)