

02291: System Integration

MUD Game Design

The task of this exercise is to create a design document containing sufficient information for a programmer to implement the system. The design document does not need to take any physical implementation into account.

- One should assume that each software component can talk to another component via a standard component protocol.
- The programmer himself chooses the right implementation language for each device and the communication technology between devices.

Furthermore, one should abstract away from any concrete user interface representation.

- Thus the user interface and its implementation is not part of the design to be described in this exercise. The focus should be on the application layer functionality.
- However, it may be helpful to draw sketches of the user interface to get a feeling for the functionality needed in the application layer.

Explain any non-trivial design decision that you make and state any assumptions that you make.¹ You can document them by

1. Describe the design issue / problem
2. Some alternative solutions and their consequences for the system
3. The selected solution

Note that the design you make should be justified by the use cases defined in the previous exercise. It does not make sense to design something that is not justified by a requirement the user has. To reduce the size of the task, you should only make a design of your system such that the use case you have chosen to detail in the previous exercise form the basis of the design. You can detail more use cases if you think they lead you to an improved design.

MUD Game overview over the system architecture

Provide a high-level view of the architecture of the system using components, ports, and connectors using a composite structure diagram or a component diagram.

¹You may want to check these assumption with me first before assuming them.

MUD Game rough system design

Use CRC cards to create a rough design of the system. Base your CRC cards session on the use cases you have detailed in the last exercise.

MUD Game Component Design

- Define the components of the system with their ports (i.e., a port is a set of required and provided interfaces).
- For each port and required and provided interface describe the protocol using a state machine.

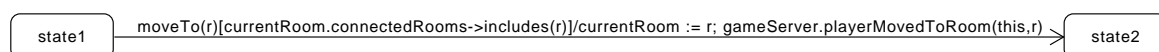
MUD Game Detailed Class Design

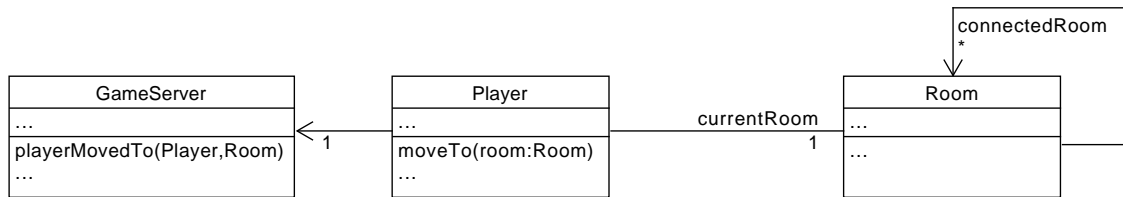
- Provide class diagrams to show how the components are implemented.
 - Some components can be implemented as a single class; other components can be implemented by a set of classes.
- There should be one class diagram for each component that is implemented by a class or a set of classes
 - Make sure that the class diagrams show the required and provided interfaces of the component and which class implements/uses the interfaces.
- There can be an additional class diagram for those classes of objects that are exchanged between the components, i.e., as parameters of methods of the required/provided interfaces.

MUD Game Behaviour Design

Describe the behaviour of each class using state machines.

- Make sure that the notation is correct. Also make sure that that you use Life Cycle Statemachines (LSMs) for classes. That means that the statemachine belongs to a class and describes its life, and that the triggers in the labels of a transition must be one of the operations of that class. For example, if you have a class Player (c.f. the excerpt of a class diagram below), which has a method `moveTo(room)`, then your LSM for class Player should contain a transition with `moveTo(room)` as the trigger, like in the following example:





Documenting the MUD game design

- Extend the report from last exercise with a section on the design of the MUD game. Thus the overall report should have the following structure:

A title page

Contains title of the report and authors

1. Requirements

1.1. Domain analysis

1.2. Functional requirements

1.3. Non-Functional requirements

2. Design

2.2. Component Design

2.3. Detailed Class Design

2.4. Behaviour Design

- **Important:** Update your requirements documentation to include the knowledge gained from the feedback of exercise one and when creating the design. Mark those parts that have changed from the first report.
 - Note, this does not mean to include design decisions in your requirements model, but to include your better/more detailed understanding of the requirements as a consequence of the feedback and doing the design.
- The PDF version of the paper should be submitted CampusNet.
 - The filename needs to be of the form sxxxxx.2.pdf (i.e. the student number of **one** of the group members, followed by an underscore, followed by 2, the number of the exercise)