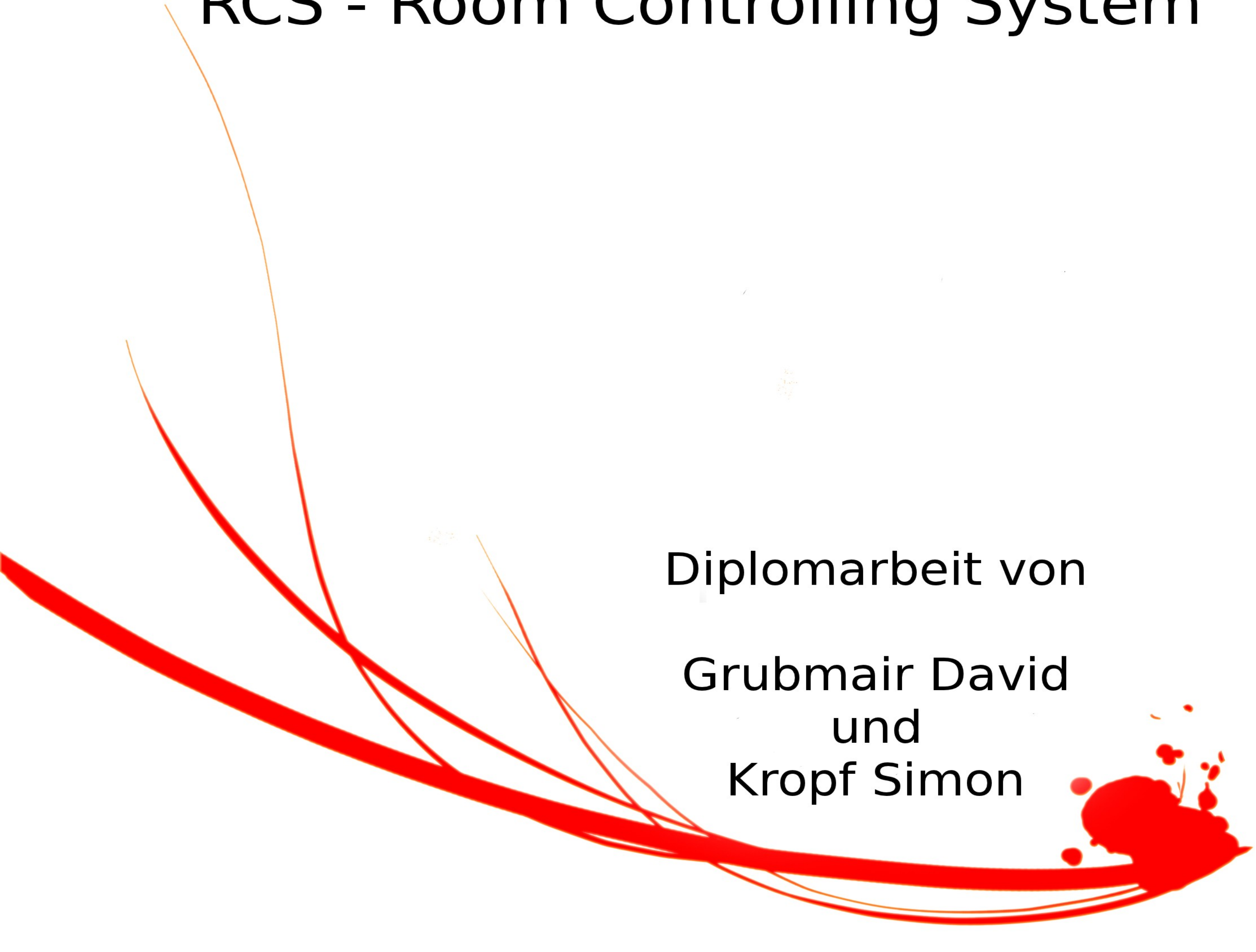
RCS - Room Controlling System



Diplomarbeit von

Grubmair David
und
Kropf Simon

1 Classification

1.1 Java EE application

The Java EE Application provides a SOAP based interface used to enable communications between devices and the Java server. The server has a socket connection to the Arduino itself which is connected through the implemented protocol named MCP (Minimal Communication Protocol). Further details about this protocol are described in the work package. A user-friendly web interface is implemented which contains display masks for displaying and processing conditions between the different actors and sensors.

For example: is it possible to trigger an entry into a room activating the alarm system.

1.2 Android application

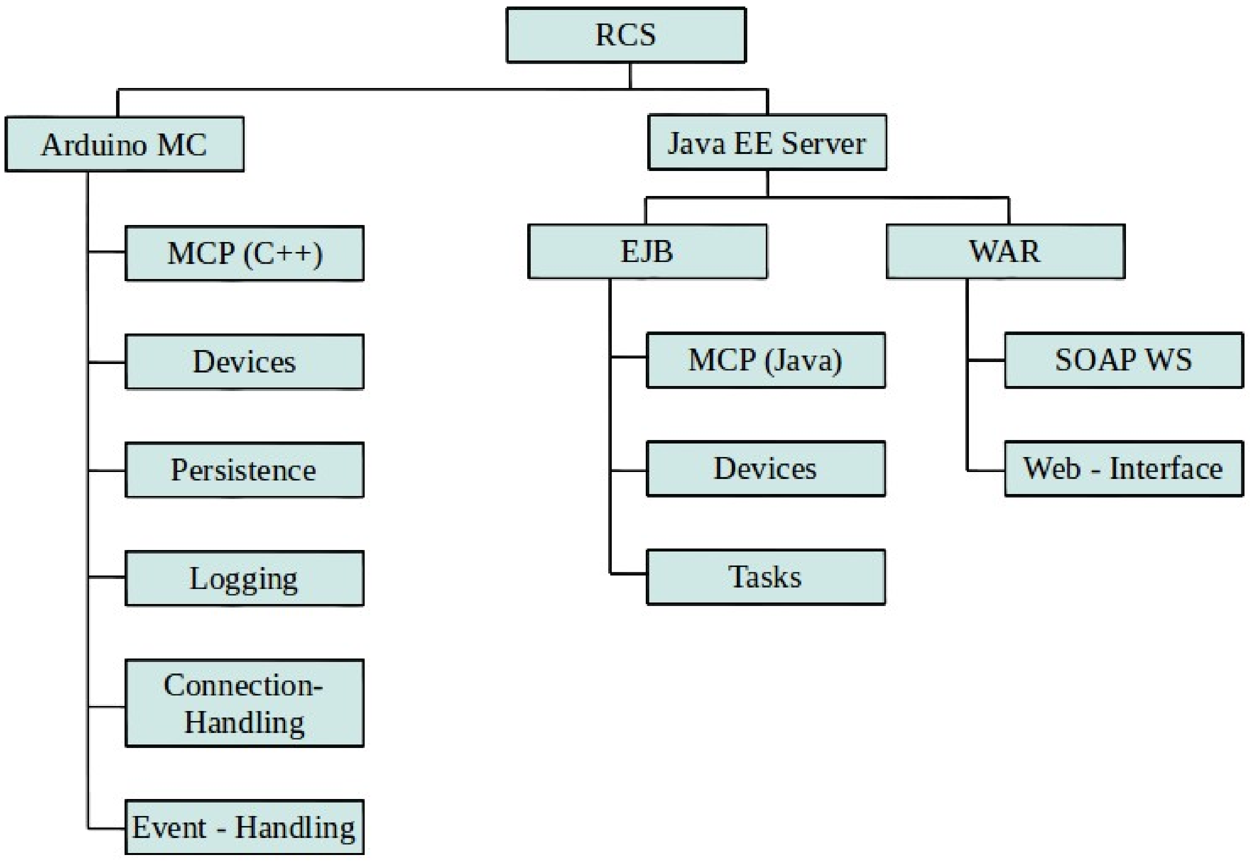
XML-Templates are provided that allow the programmers the creation of simple but dynamic User-Interfaces. A sample application will be created, showing the possibilities of these templates. It will be designed as an end-user application with only a view possible different settings keeping it as simple as possible.

1.3 Arduino

On the Arduino side, the control and the administration of different actors and sensors is implemented. These IO devices will be isolated to complex devices which itself are able to manage themselves properly. Communication between the Java EE server and the Arduino just as between the Android - application and the Arduino is handled by a specially developed protocol named “MCP”. It is described in the eponymous working packet. Each device, actor and sensor are logged and persisted on the microcontroller.

Logging can be done in the following ways: - SD Card - UDP - Serial

2 Project structure plan



JAVA EE Server

EJB

MCP (Java)

This is the “Minimal Communication Protocol” (MCP), which is responsible for the communicationbetween the Java Server / Android Device and the Arduino MC. This work package containsfollowing classes:

- CallMessage	- MCPServer
- LogMessage	- MCPCommunicator
- DeviceMessage	- MCPListener
- ResultMessage	- MCPWriter
- GlobalMessageID	

There are four different message types.

CallMessage	= call a method
LogMessage	= to log something
DeviceMessage	= represents a device
ResultMessage	= returns one or more results from a call

The GlobalMessageID is a system-wide identification number, which is used for a uniqueidentification of a message.

The last four classes are the administration of this protocol. The MCPServer waits for a incomingconnection and starts the MCPListener for this connection. In the MCPListener the incomingmessages are worked off. The MCPWriter sends a message to the Arduino MC. At last theMCPCommunicator which is responsible for the interaction between the WebInterface and the MCP.

Devices

Each device has his own class with the appropriate methods. It also contains an ID which is assigned by the Arduino MC. Also the devices are provided by the SOAP WS for external usage.
The devices are persisted at the Java Server for statistic usage, but the essential information ispersisted at the Arduino MC.

WAR

SOAP WS

The SOAP WebService is used to communicate with an Android / SOAP – compatible devices. Thefeature of this WS is that it only has one method called “getDevice”. As parameters it has the nameor the id of the device.
Because it is using a technology called SEI (Service Endpoint Interface), it return a reference of thedevice. And now if a method of the device is called at the client, this call is sent over to the JavaServer and will be called there.

Web – Interface

The Web-interface is made up of JSF, CSS and JAVASCRIPT.
It's able to control any amount of Arduino MCs.
The devices are loaded dynamically and therefore it's expendable at will.
It uses the MCPCommunicator and various device entities to communicate with the selectedArduino MC.

Arduino MC

Persistence

Loading, saving from the SD - Card and managing the devices is done here.This work package contains following classes:

- DeviceManager
- IPersistable
- UnifiedFactory

<u>DeviceManager</u>
Holds a list of devices, actors, sensors, IOActors and IOSensors available in the system. It provides Methods to load and to save this devices.

<u>IPersistable</u>
Every class which wants to be persisted with the DeviceManager needs to implement this interface.

<u>UnifiedFactory</u>
Every persisted class can be dynamically loaded as new instances in this Factory. Classes can only be loaded when they implement IPersistable.

MCP (C++)

This is the “Minimal Communication Protocol” (MCP), which is responsible for thecommunication between the Java Server / Android Device and the Arduino. This work package contains following classes:

- MCP_Message
- Result_Message
- Log_Message
- Device_Message
- Call_Message
- Global_MessageID
- MCP_Interpreter

There are four different message types.

CallMessage	= call a method
LogMessage	= to log something
DeviceMessage	= represents a device
ResultMessage	= returns one or more results from a call

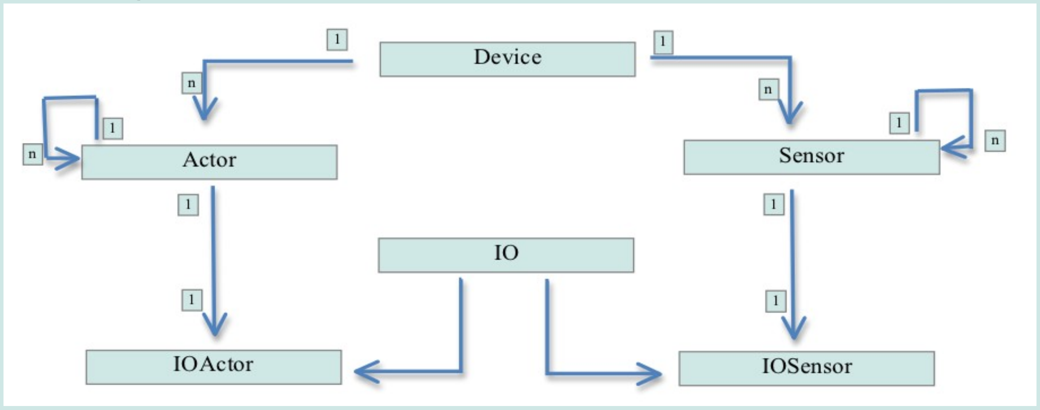
The GlobalMessageID is a system-wide identification number, which is used for a unique identification of a message.

The last class is for the administration of this protocol. It interprets the String message and creates the specific Call Message for the Device – Method – Calls.

Devices

Each device has his own class with the appropriate methods. It also contains an ID which is assigned by the Arduino MC. Also the devices are provided by the SOAP WS for external usage. The devices are persisted at the Java Server for statistic usage, but the essential information is persisted at the Arduino MC.

Simplified class diagram:



Logging

There are two ways of Logging.

Every class can Log if they want to log with the static class Log. There are three ways of Logging such events classified by the status.

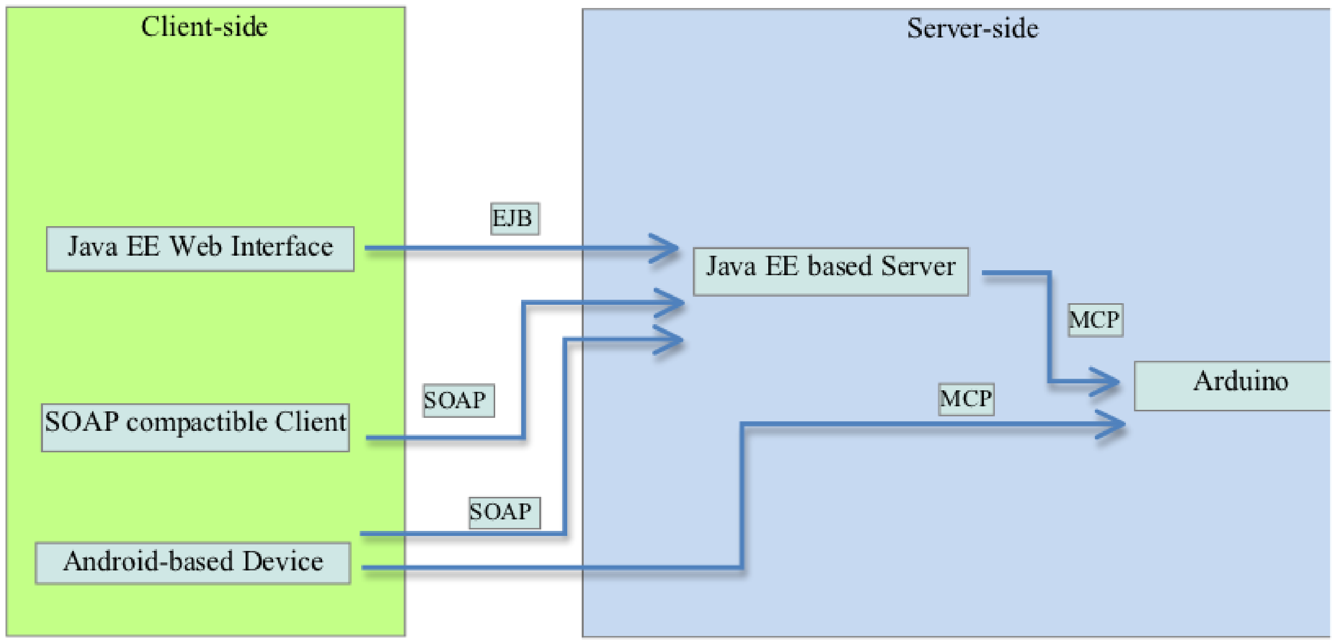
- Verbose (info)
- Error
- Warning

The second way is that every class can implement the interface ILogable and then the status of the class can be retrieved.

Connection – Handling

Any incoming connection will be managed by the Connector class. It provides methods and callbacks activated when connections are established. Four connections can be handled at the same time.

3 System architecture


Client-side
Server-side
Java EE Web Interface
EJB
Java EE based Server
MCP
Arduino
SOAP compactible Client
SOAP
MCP
SOAP
Android-based Device