Component-Based Software Engineering	**Exercise Sheet No. 10**
Dipl.-Medieninf. Christian Piechnick	Software Technology Group
INF 2080	Institute for SMT
`christian.piechnick@tu-dresden.de`	Department of Computer Science
	Technische Universität Dresden
	01062 Dresden

Role-Based Design and Programming

Task 1: Role-Based Design

1a) In the lecture it was explained that roles are non-rigid and founded. Explain those terms.

1b) Draw the class-diagram of the Role-Object Pattern and explain the intention.

1c) Differentiate the terms role, role-type, role-type diagram, class-role-type diagram.

1d) The lecture claims, that roles seperate concerns. Explain this property.

1e) What role constraints do you know? Explain them.

Task 2: Role-Based Programming with ObjectTeams

In this task we will use the Role-based Programming (ROP) language ObjectTeams to implement some features to a robot simulation environment. The environment consits of machines producing products and robots able to transport those products to certain positions. The simulator takes orders, containing several products. In every cycle, the simulator takes an available robot (placed at the charging station) and sends the robot to a machine. After the robot arrived the machine, it produces a product and places the product on the robot. The robot carries the product to the storage, drops it there and drives back to the charging station. The simulation also includes the robot's battery. Every 100ms, the battery level (percentage from 0 - 100) decreases by 1 percent. Furthermore each robot has a maximum load weight. When a product is placed on the robot, which is heavier than this maximum load, the robot breaks. In addition, the simulator also models persons, having a qualification (i.e., manager or engineer).

2a) Download ObjectTeams (`http://www.eclipse.org/objectteams/download.php`)

2b) Work through the documentation and implement some examples (`http://wiki.eclipse.org/Category:Object_Teams`)

2c) Download the RobotSimulation Eclipse project, available on the exercise website and get the project running.

2d) Currently, the simulator will always use the maximum speed of the robots to drive within the production hall. However, when the robot carries products, the maxium speed shall limited to 750 (mm/s) in order to avoid damages. Implement a `SpeedLimiterRole` that limits the maxium speed of a robot to 750, when it is carrying products.
Tip: Use callins to replace the previous behaviour when the `driveTo(Position p, int speed)` method of the robot is called.

$\boxed{\text{2e)}}$ Currently, the simulation always fails, because the battery of all robots will drop to zero, before the order is successfully served. Implement a team `Charging` which adds the robot to the list of the charged robots (i.e., `ChargingStation.charge(Robot r)`) when the corresponding robot is placed at the charging station. Furthermore, the robot should be removed from the list of charged robots, when it leaves the charging station.
Tip: Use the `setCurrentPosition(Position pos)` method of the class `Robot` to detect when the robot is placed at the charging station.

$\boxed{\text{2f)}}$ Extend the Charging team by a `CriticalBattery` role, which is played, when the batteryLoad is less then 40 percent. When the robot plays the CriticalBattery role, it is not allowed to take any tasks.

$\boxed{\text{2g)}}$ Currently, all TurtleBots will break because the simulator will instruct them to carry to heavy objects. Implement a role `LoadWatcher` refusing tasks to carry objects, which are heavier then the maximum load of the corresponding robot.

$\boxed{\text{2h)}}$ Implement a role that counts the number of products, produced by a machine.

$\boxed{\text{2i)}}$ Now, implement a role that marks a machine as broken, when the number of produced products is greater then 5.

$\boxed{\text{2j)}}$ Implement a role `MaintainerWorker` that is a assigned to a broken machine and will repair it, when his `work()` method gets called.

1