AS-116.3200 Real-time system modeling
Exam 9.3.2006 (max 36 p)

1) UML modeling language is based on a *metamodeling approach*. UML specification defines the notions of the language at different levels of abstraction. (6p)
a) How many layers does this metamodeling approach have? At which of these layers situates the *UML –metamodeling language* itself and at which of these layers situates the user defined application model?
b) Inside one layer new element notions are defined as specializations of the more general elements (using *subClassOf* relationship). What is the relationship between elements at different metamodeling layers (relationship over the layer border) ?

2) UML provides means to model concurrency at several levels in a model also in states of a state diagram. In this example case the *PlatformSystem* of a special robot vehicle is responsible of controlling robot motion, its driving and steering. You should refine one state of the state diagram model of the system. (6p)
PlatformSystem inherits its state diagram from its superclass *Platform* (see fig.1) You should refine the *Driving* superstate so that both wheel *driving* control and *steering* control could be done concurrently. The state behavior is specified as follows:
1: Initially when coming to the *Driving* superstate the control of these two subfunctions is independent from each other and concurrent.
2: an external event, *evCoordinated*, should change the operation mode to *CoordinatedDrive* mode, in which the both subfunctions, driving and steering, are coordinated and synchronized together for more efficient motion control.
3: an external event *evSimpleDrive* should trigger the system out of the *CoordinatedDrive* mode and system should return to the state where it came initially (in step 1:)
4: an external event *evInterrupted* should trigger a state change out of the *Driving* state to the *Interrupted* state (see fig. 1)
5: an external event *evResume* should trigger a state change back to the *Driving* state and into the same subState(s) where it was while ev*Interrupted* event occurred.

You should draw the sub state diagram only for the *Driving* superstate and only one nested state level (e.g. do not draw the inner behaviour state machine of the state *CoordinatedDrive*). You should use *pseudo-states* when ever possible or appropriate.

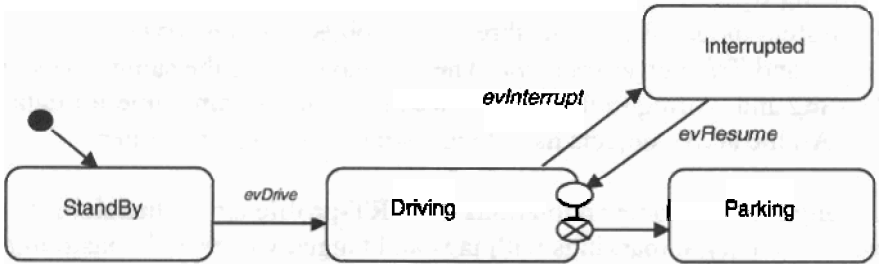

Figure 1: State diagram for the *Platform*

3)
Which UML diagrams are especially suitable for presenting user and system requirements and facilitate communications between end-users and developers about the system? What is the purpose of each of these diagrams and how do they relate to each other? (6p)

4)
Illustrate the following concepts (using the UML2 class diagram notation) with any work machine of your choice:
- structured classes,
- ports,
- interfaces and contracts,
- connectors.

Your machine can be simplified, even unrealistic. It does not need to take into account all exceptional situations. Just specify some set of functionality that is enough for illustrating these concepts. Give an explanation of one operation and how the elements in your model are involved in processing that operation. (6p)

5)
a) Describe with a few sentences what does the concept, *Design Pattern*, mean in the object oriented programming context. What are the most important parts of the DP description? (3p)

b) Describe the idea, structure and functioning of EITHER the *Monitor-Actuator Pattern* OR the *Observer Pattern* (a.k.a Publish Subscribe pattern). Draw also some kind of diagram of the collaborating objects or classes for support of your explanation. (3p)

6)
This question relates to *Schedulability, Performance and Time* profile, also called as RT-profile. The following figure 2 presents a UML communication diagram of one example model case. The system modeled contains three active objects *TelemetryGatherer*, *TelemetryDisplayer* and *TelemetryProcessor*. They all have to use the same data storage, *SensorData*. Reading and writing to it is not allowed to do at the same time for data integrity reasons. All the active objects have their own Clock used as a timer.

Add to the following figure 2 some annotations from RT-profile e.g. Schedulability annotations; that is: stereotype markings with tags and tagged values and constraints, that could be used in this kind of model and diagram for the schedulability analysis purposes. The **appendix A** contains a table listing several stereotypes and related tags, from which you may apply the appropriate ones. (6p) (Note Figure 2 at the next page)