AS-116.3200 Real-time system modeling
Exam 6.3.2006 (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 and what 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 (i.e. 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. In this exercise you should refine one state of the state diagram model of the system and apply appropriate *pseudo-states* (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 for the *Driving* super-state that implements the above described behavior and conforms to the *Platform* state model in Fig 1. Only one nested state level is needed (e.g. do not draw the inner behaviour state machine of the state *CoordinatedDrive*). Note that you should use *pseudo-states* when ever possible or appropriate.

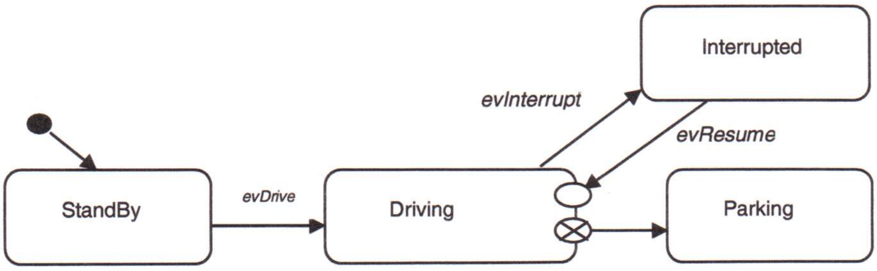

Figure 1: State diagram for the *Platform*

3) The following concepts are related to Requirement management. Explain briefly these concepts or things.

a) Measurability of non functional requirements. (1p)
b) User requirements (1p)
c) System requirements. (1p)
d) Requirement traceability (1p)
e) Requirement. engineering process (1p)
f) Scenario in requirement engineering (1p)

4)
UML2 structural modeling provides ports for communication between classes. How can ports be utilized to improve the reusability of software?
Compare this to using references to objects as taught in the C++ or Java basic programming course. Use an example to explain your answer. (Drawing a UML diagram is optional. A clearly written textual answer is acceptable.) (6p)

5)
a) Describe with a few sentences what does the concept, *Design Pattern*, (Suunnittelukaava) 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). What are the benefits of using this pattern? Draw also some kind of diagram of the collaborating objects or classes for support of your explanation. (3p)

6)

Object Constraint Language (OCL) specification is one of the UML 2.0 specifications. Describe the meaning of *constraint* notion in UML modeling context. Describe also what is OCL? Explain why it is used and how it is applied in UML models? (6p)