Prof. Dr. Detlef Kips, Dr. Martin Jung
Department Informatik, Universität Erlangen-Nürnberg

31.03.2017

Klausur: Praktische Softwaretechnik

Personal Information (Please use block letters!):

Name, Given Name:	_____
Date of birth:	_____
Matrikelnummer:	_____

Please read the following hints and instructions carefully and confirm your understanding by signing below!

- Please put your student ID and a personal identification (with a picture of you) on the desk and have it ready for check of identity and participation!
- **No tools or resources besides writing equipment are allowed!**
- The solution must be given in the space provided. If the space is not enough for your solution, use the additional space at the end of this printout. Provide a short reference to the pages in the back at the original task. You may ask for additional empty pages, they will be provided by the supervisors and will be stapled to the printout.
  You will only be allowed to hand in the stapled printout.
- You may request scrap paper from the supervisors.
  These may not be handed in after the test.
- The working time is 90 minutes. You can reach 90 points.
- If you can't continue the test for health reasons, you have to prove this by providing a *„Erweitertes ärztliches Attest"* to the Prüfungsamt. In this case, signal the supervisor, request the corresponding form, and fill it out before leaving the test.
- **Check the printout for completeness! The printout has 13 pages, and an additional page with task information (not stapled). Check also if the print is readable.**

By my signature, I confirm the receipt of a complete test printout, and acknowledge the reading and understanding of the information given above.

Erlangen, den 31.03.2017 .....

(Signature)

This will be filled out by the teacher, NOT by the student!

Score:

Task	1	2	3	4	5	Σ
Score						

Grade:

In the following:

- Multiple-Choice-Tasks (1.a to 1.g), will be awarded at least 0 points. Generally, each error will result in deduction of a point. In particular, this means, that missing check marks are considered to be errors! In each sub-task (a-g), at least one statement is true, and you have to check all true statements.
- In your answers, especially when drawing diagrams, mind the **readability**. If we can't read the answer, we have to consider it wrong.
- In all tasks, only the final result is relevant, intermediate steps should be left out or put on scrap paper.
- We **strongly suggest** you to draft all diagrams on scrap paper and transfer the final result as a readable diagram to the printout.

Task 1 [21 points]

a) The Unified Modeling Language ...

- ☐ supports the whole development life cycle.
- ☐ enforces the usage of a certain development process.
- ☐ implies the usage of a certain modeling notation.
- ☐ implies the usage of a certain modeling methodology.
- ☐ enforces the usage of a certain programming language.
- ☐ enforces the usage of a certain tooling environment.

b) The external behavior of a use case's default scenario may be specified ...

- ☐ using a use case diagram.
- ☐ as a generic scenario.
- ☐ as an exemplary scenario.
- ☐ using an activity diagram.
- ☐ using a sequence diagram.
- ☐ using an entity relationship diagram.

c) A certain bug in program code may be identified and fixed ...

- ☐ during a code review
- ☐ using code metrics
- ☐ during module testing
- ☐ during white box testing
- ☐ during black box testing
- ☐ during a design review

d) A Gantt Chart ...

- ☐ shows duration of work packages.
- ☐ is used only during estimation sessions.
- ☐ shows the logical sequence of work packages.
- ☐ shows the project organization.
- ☐ is used to represent concurrent workflow among components.
- ☐ contains resource allocation.

e) The following activities are part of the coding workflow:

- ☐ creating detailed design specification.
- ☐ reviewing finished program code of team members.
- ☐ modeling the code deployment.
- ☐ documenting source code.
- ☐ refining non-functional requirements.
- ☐ checking in meaningful functionality.

f) A software architecture model contains …

- ☐ functional requirements.
- ☐ dependencies among the components.
- ☐ control flow instructions for methods.
- ☐ representations of physical files.
- ☐ processor architecture.
- ☐ network connections and their attributes.

g) Design Patterns …

- ☐ are specific solutions for functional requirements.
- ☐ need to be adapted to the context of their application.
- ☐ can describe the creation of objects.
- ☐ are not re-usable outside of their context.
- ☐ contain collaborations showing other patterns they interact with.
- ☐ are typically used for intra-component design.

Task 2 [19 points]

a) Enumerate 4 essential properties of a good requirements specification!

b) Name three types of UML Diagrams you could use in order to specify the precise dynamic and timing behavior of a technical system like an automotive control unit!

c) Enumerate four typical review variants, sorted by increasing effort of preparation and execution!

d) Considering an agile software development project, name an example for each of the following process elements!

Artifact:

Role:

Activity:

e) Name three architecture views according to Kruchten!

Task 3 [25 points]

Imagine your development team has to realize a traffic light controller software. Your task is to elaborate an analysis model covering (among others) the following requirements:

REQ.1. The Traffic Light System consists of four Car Traffic Lights (**CTLs**) and two Pedestrian Traffic Lights (**PTLs**).

REQ.2. When a pedestrian will press the push-button at one of the **PTLs** while the **CTLs** being in state green, the **CTLs** have to change their states to yellow. While the **CTLs** are in state yellow or in state red, a pedestrian's request will be ignored.

REQ.3. A **CTL** stays in state yellow for a time period **t0**, then it changes its state to red.

REQ.4. After the **CTLs** having changed their states to red, the **PTLs** have to wait for a delay time of at least **t1** before changing their states to green.

REQ.5. A **PTL** stays in state green for a time period **t2**, then it changes its state to red again.

REQ.6. After the **PTLs** having changed their states to red, the CTLs have to wait for a delay time of at least **t3** before changing their states to green again.

REQ.7. If a pedestrian's request comes in while the **CTLs** being in state green, the request has occasionally to be delayed until the **CTLs** have been in state green for a time period of at least **t4**. After this delay time, the pedestrian's request will be dealt with according to *REQ.2*.

During a CRC Card Session, your team colleagues identified the class candidates:
`CarTrafficLight`, `PedestrianTrafficLight`, `TrafficLightController`.

a) Draw a UML class diagram for the Traffic Light System, using these class candidates and specifying them with attributes, operations, associations, multiplicities and gen/spec-relationships!

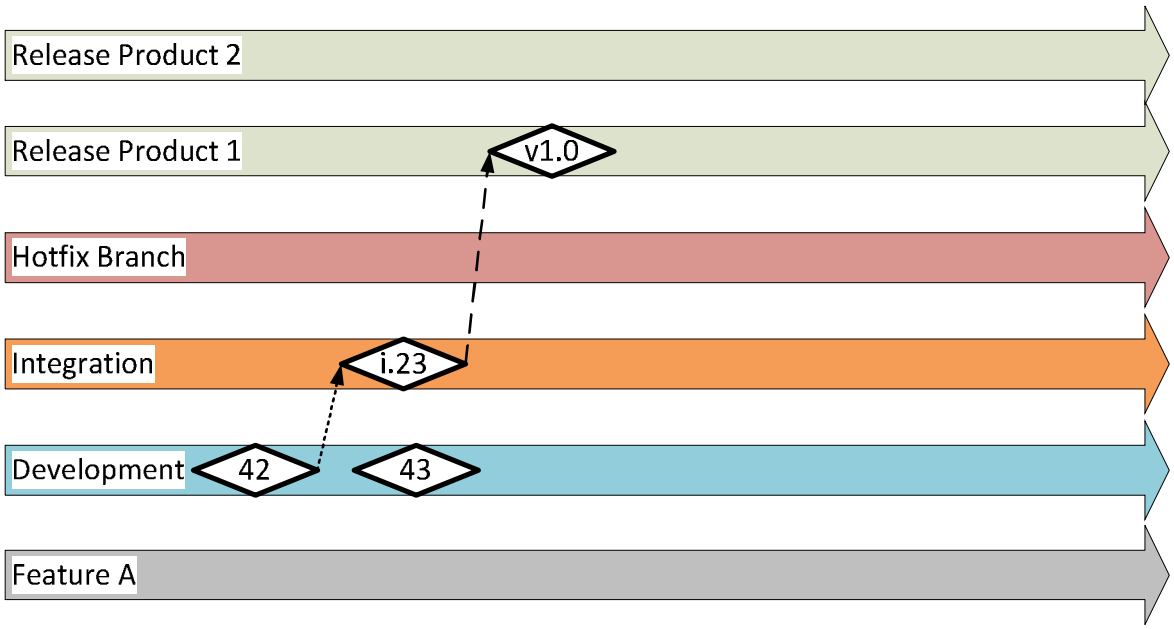
b) For the default scenario of the use case "Pedestrian Request", draw an exemplary internal scenario representation using an UML sequence diagram! For this exemplary scenario, assume that when the pedestrian's request comes in the **CTLs** have been in state green for a time period longer than **t4**. The scenario ends, when the **CTLs** are in state green again.

c) Draw a UML state diagram representing the object life cycle of the TrafficLightController Object!
Hint: In order to find appropriate states for the controller object, you should combine the state information of the **CTLs** and the **PTLs** to a current state information of the overall system configuration.

Task 4 [10 points]

Given the following branch model, draw the following merge edges and use meaningful names for the resulting versions:

a) Start development of feature A from version i.23.
b) Fetch a bug reported on Product 1 v1.0 to the hotfix branch, fix it, and push a version v1.0.1 back in the field.
c) Merge the bug-fix back into development, develop a clean solution and pull this version on the integration branch.
d) After three development sprints, feature A is ready for deployment, pull it on the integration branch.
e) Create a product 2 v1.0 that contains the new feature and all known bug fixes.



Task 5 [15 points]

You are the project manager of a traditional software development project.

Your team consists of the following people with their respective skills

- *Daniel*: software requirements engineering and software architecture
- *Huey* and *Louie*: software development and unit testing
- *Dewey*: software integration
- *Donald*: software testing

You plan for the following activities (estimated effort [days] in parantheses):

A. Requirements Analysis (3)
B. Software Architecture (5)
C. Development Component A (5)
D. Development Component B (7)
E. Unit Testing A (6)
F. Unit Testing B (7)
G. Integration (5)
H. Software Test (5)

a) Draw a Gantt chart and mark the critical path.
b) Calculate the shortest time this project will take.
c) During development, you discover that C is more complicated. You and your team re-estimate C, and conclude that its effort is 10 days. How long is the critical path?

Shortest time of project (5.b):

Length of critical path after re-estimation (5.c):