SI4-KBS
Component-based Software Engineering
Oral examination

Time and place

Exam: June 7-10, 14-17, 2021.

Re-exam: August 19, 2021.

Examination

The format of the examination is a **20 minutes** oral exam based on a portfolio. The exam questions are categorized by topic and published in good time before the examination. It is recommended that the student prepares a short presentation based on the exam topics. It is important that the presentation material used in the presentation is prepared by the individual student. That is, it is the student's own understanding of the material that is being used for the further evaluation.

The result of the examination is graded according to the *7-point scale*. During the exam, the student must demonstrate any skills and understanding of the principles and concepts, related to the topic to which the question relates. The student should strive to provide a fairly comprehensive description of the topic and to demonstrate skills through examples (Portfolio). Feel free to use the course assignments as example material.

The students have to bring their own computer that can be used for the presentation.

Topics

1. Components Models in Java: "What are the different types of component models?", "What is the Spring framework?", "What are Spring beans and the Spring container?", "What is the significance of the Spring configuration?", "What is OSGi?", "What is a bundle in OSGi?", "What is the importance of the OSGi service registry?", "What is MANIFEST.MF in OSGi?", "What is component.xml in OSGi?", "How are the components created using Netbeans, Spring, OSGi and invoked by an application client?" and "Explain the configuration mechanism available in the component models".

2. Component-Oriented Design and Architecture 1) What are the different types of component models?, 2) What is tight coupling?, 3) Why is tight coupling not advisable? 4) How do we measure the amount of coupling present in a design?, 5) What design strategies are used to lower coupling?, 6) Why should the business entity objects not be offered as dynamic component services?, 7) Why is coupling on a stable entity preferred over coupling on an unstable entity?,
8) Should componentization of multiple layer applications preserve the original layering structure? and, 9) When changes are requested, is it advisable to change the provided and required interface definitions?

3. JDK Service Loader 1) What are the basic component structural constructs in Java SE platform?, 2) What are the pros and cons of basic component constructs provided by Java SE platform? 3) Why is a component interface important? What is the primary reason for introducing a component interface?, 4) What is glue code? Why is it required?, 5) What is the alternative to using glue code?, 6) What is a component model?, 7) What are the different types of component models?, 8) What is the relationship between a component model and a component framework?, 9) How are components wired in a whiteboard component framework?, and 10) Can a component with the provided interface take the responsibility of registering it with a whiteboard component registry?

4. OSGi 1) What is wrong with the JAR file based component model?, 2) How does OSGi overcome the problems of flat CLASSPATH?, 3) Are all OSGi bundles JAR files?, 4) Are all JAR files OSGi bundles?, 5) How is the unique identity of an OSGi bundle defined?, 6) Do all OSGi bundles need to export one or more packages?, 7) Can an OSGi bundle transition its state from installed to active?, 8) How does the OSGi framework carry out bundle resolution? Is the same resolution mechanism applicable to the OSGi service layer?, 9) What are the elements of an OSGi service?, 10) How do you register an OSGi service?, 11) How do you find an OSGi service?, 12) How do you bind to an OSGi service?, 13) Why do we need declarative services?, 14) Can an OSGi component provide more than one interface?, 15) Can an OSGi component require more than one interface?, 16) Do declarative services provide auto-wiring of required and provided interfaces?, and 17) In what way does declarative services specification enhance upon the service layer specification?

5. Spring 1) What is a Spring container?, 2) What is a Spring bean?, 3) What is Spring configuration and what options exists?, 4) What is the two types of Spring beans?, 5) How are two Spring beans wired automatically by the container?, and 6) How are the business components in Spring represented and identified?,

6. Netbeans 1) What is wrong with the JAR file–based component model?, 2) How does Netbeans overcome the problems of flat CLASSPATH?, 3) Are all Netbeans modules JAR files?, 4) Are all JAR files Netbeans modules?, 5) How is the unique identity of a Netbeans module bundle defined?, 6) Do all Netbeans modules need to export one or more packages?, 7) How does the Netbeans runtime-container carry out module resolution?, 8) What are the elements of a Netbeans service?, 9) How do you register a Netbeans service?, 10) How do you find a Netbeans module service?, 11) Can a Netbeans module provide more than one interface? How?, 12) Can a Netbeans module require more than one interface? How?,

Syllabus

[BBB$^+$00] Felix Bachmann, Len Bass, Charles Buhman, Santiago Comella-Dorda, Fred Long, John Robert, Robert C Seacord, and Kurt C Wallnau. Volume II: Technical Concepts of Component-Based Software Engineering, 2nd Edition. II(May):65, 2000. [Link].

[Boc11] Heiko Bock. *The Definitive Guide to NetBeans^{TM} Platform 7.* Apress, 2011. [Link].

[Fow04] Martin Fowler. Inversion of Control Containers and the Dependency Injection pattern, 2004. [Link].

[Lora] Richard Lord. What is an entity system framework for game development? [Link].

[Lorb] Richard Lord. Why use an entity system framework for game development? [Link].

[MBN68] M. D. McIlroy, J. M. Buxton, and P. Naur. Mass-produced software components. Conference proceedings (article), NATO Science Committee, Garmisch, Germany, January 1968. [Link].

[MSS13] P. Manickam, S. Sangeetha, and S.V. Subrahmanya. *Component-Oriented Development and Assembly: Paradigm, Principles, and Practice using Java.* Infosys Press. Taylor & Francis, 2013. [Link].

[Neta] Netbeans. Part 1: Runtime Container. [Link].

[Netb] Netbeans. Part 2: Lookup API. [Link].

[Ora] Oracle. Trail: The Extension Mechanism. [Link].

[Szy97] Clemens Szyperski. *Component Software: Beyond Object-Oriented Programming.* Addison-Wesley Professional, December 1997. [Link].

[Tul] Jaroslav Tulach. Lookup and Spring. [Link].