Exercises
Engenharia de Software
(cod. 5386 & 6633 )

Departamento de Informática
Universidade da Beira Interior
Ano lectivo 2010/2011

These exercises are taken from Software Engineering, 9th edition,

Simão Melo de Sousa

Chapter 1. Introduction to Software Engineering

1. What is the most important difference between generic software product
development and custom software development? What might this mean
in practice for users of generic software products?

2. What are the four important attributes that all professional software
should have? Suggest four other attributes that may sometimes be signif-
icient.

3. Apart from the challenges of heterogeneity, business and social change and
trust and security, identify other problems and challenges that software
engineering is likely to face in the 21st century (hint: think about the
environment).

4. Based on your own knowledge of some of the application types discussed
in section 1.1.2, explain, with examples, why different application types
require specialized software engineering techniques to support their design
and development.

5. Discuss whether professional engineers should be certified in the same way
as doctors or lawyers.

Chapter 2. Software Processes

1. Giving reasons for your answer based on the type of system being devel-
oped, suggest the most appropriate generic software process model that
might be used as a basis for managing the development of the following
systems:
• A system to control anti-lock braking in a car
• A virtual reality system to support software maintenance
• A university accounting system that replaces an existing system
• An interactive travel planning system that helps users plan journeys with the lowest environmental impact

2. Consider the reuse-based process model shown in Figure 2.3. Explain why it is essential to have two separate requirements engineering activities in the process.

3. Suggest why it is important to make a distinction between developing the user requirements and developing system requirements in the requirements engineering process.

4. Explain why change is inevitable in complex systems and give examples (apart from prototyping and incremental delivery) of software process activities that help predict changes and make the software being developed more resilient to change.

5. What are the advantages of providing static and dynamic views of the software process as in the Rational Unified Process?

Chapter 3. Agile Software Development

1. Explain how the principles underlying agile methods lead to the accelerated development and deployment of software.

2. When would you recommend against the use of an agile method for developing a software system?

3. Extreme programming expresses user requirements as stories, with each story written on a card. Discuss the advantages and disadvantages of this approach to requirements description.

4. Suggest four reasons why the productivity rate of programmers working as a pair might be more than half that of two programmers working individually.

5. It has been suggested that one of the problems of having a user closely involved with a software development team is that they adopt the outlook of the development team and lose sight of the needs of their user colleagues. Suggest three ways how you might avoid this problem and discuss the advantages and disadvantages of each approach.

Chapter 4. Requirements Engineering

1. Discover ambiguities or omissions in the following statement of requirements for part of a ticket-issuing system:
An automated ticket-issuing system sells rail tickets. Users select their destination and input a credit card and a personal identification number. The rail ticket is issued and their credit card account charged. When the user presses the start button, a menu display of potential destinations is activated, along with a message to the user to select a destination. Once a destination has been selected, users are requested to input their credit card. Its validity is checked and the user is then requested to input a personal identifier. When the credit transaction has been validated, the ticket is issued.

2. Write a set of non-functional requirements for the ticket-issuing system, setting out its expected reliability and response time.

3. Suggest how an engineer responsible for drawing up a system requirements specification might keep track of the relationships between functional and non-functional requirements.

4. Using your knowledge of how an ATM is used, develop a set of use cases that could serve as a basis for understanding the requirements for an ATM system.

5. When emergency changes have to be made to systems, the system software may have to be modified before changes to the requirements have been approved. Suggest a model of a process for making these modifications that will ensure that the requirements document and the system implementation do not become inconsistent.

Chapter 5. System Modelling

1. How might you use a model of a system that already exists? Explain why it is not always necessary for such a system model to be complete and correct. Would the same be true if you were developing a model of a new system?

2. Develop a sequence diagram showing the interactions involved when a student registers for a course in a university. Courses may have limited enrollment, so the registration process must include checks that places are available. Assume that the student accesses an electronic course catalog to find out about available courses.

3. Look carefully at how messages and mailboxes are represented in the email system that you use. Model the object classes that might be used in the system implementation to represent a mailbox and an email message.

4. Based on your experience with a bank ATM, draw an activity diagram that models the data processing involved when a customer withdraws cash from the machine.

5. You are a software engineering manager and your team proposes that model-driven engineering should be used to develop a new system. What factors should you take into account when deciding whether or not to introduce this new approach to software development?
Chapter 6. Architectural Design

1. When describing a system, explain why you may have to design the system architecture before the requirements specification is complete.

2. Explain why design conflicts might arise when designing an architecture for which both availability and security requirements are the most important non-functional requirements.

3. Explain how you would use the reference model of CASE environments (available on the book's web pages) to compare the IDEs offered by different vendors of a programming language such as Java.

4. Using the generic model of a language processing system presented here, design the architecture of a system that accepts natural language commands and translates these into database queries in a language such as SQL.

5. Using the basic model of an information system as presented in Figure 6.16, suggest the components that might be part of an information system that allows users to view information about flights arriving and departing from a particular airport.

Chapter 7. Design and Implementation

1. Using the structured notation shown in Figure 3 of chapter 7, specify the weather station use cases for Report status and Reconfigure. You should make reasonable assumptions about the functionality that is required here.

2. Using the UML graphical notation for object classes, design the following object classes, identifying attributes and operations. Use your own experience to decide on the attributes and operations that should be associated with these objects.
   (a) a telephone
   (b) a printer for a personal computer
   (c) a personal stereo system
   (d) a bank account
   (e) a library catalogue

3. Draw a sequence diagram showing the interactions of objects in a group diary system, when a group of people are arranging a meeting.

4. Using examples, explain why configuration management is important when a team of people are developing a software product.

5. A small company has developed a specialized product that it configures specially for each customer. New customers usually have specific requirements to be incorporated into their system, and they pay for these to be developed. The company has an opportunity to bid for a new contract, which would more than double its customer base. The new customer also
wishes to have some involvement in the configuration of the system. Explain why, in these circumstances, it might be a good idea for the company owning the software to make it open source.

Chapter 8. Software Testing

1. Explain why testing can only detect the presence of errors, not their absence.

2. You have been asked to test a method called “catWhiteSpace” in a “Paragraph” object that, within the paragraph, replaces sequences of blank characters with a single blank character. Identify testing partitions for this example and derive a set of tests for the “catWhiteSpace” method.

3. What is regression testing? Explain how the use of automated tests and a testing framework such as JUnit simplifies regression testing.

4. Write a scenario that could be used to help design tests for the wilderness weather station system.

5. What do you understand by the term “stress testing”? Suggest how you might stress test the MHC-PMS.