How to contribute to the joint course on software engineering by case studies

Zoran Budimac, Klaus Bothe

Abstract. This document describes where and how the joint course on software engineering [1] uses case studies in the course and in assignments. Therefore, this document also serves as an instruction on how to build new case studies (which activities should be performed and which documents produced) in order to successfully replace the existing case studies.

1 Introduction

The Joint Course on Software Engineering (JCSE) [1] currently uses the following case studies throughout the course and in some assignments:

- the main one, used in 13 topics of the course,
- the supporting one, used in 4 topics of the course,
- additional ones, used in individual topics and in assignments.

Currently, 
- the main case study is ‘Seminar Organization’, taken from [2]
- the supporting one is ‘XCTL’
- additional ones are local (i.e. are not used in more than one topic or assignment, but still pretty important) and will not be separately discussed in this text.
<table>
<thead>
<tr>
<th>Topics</th>
<th>Main (the number of slides)</th>
<th>Supporting (the number of slides)</th>
<th>Additional (the name of the example)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. What is software engineering</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Quality criteria for software products</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. Software process models</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Basic concepts for software development documents</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. Results of the “analysis and definition” phase</td>
<td>24</td>
<td>1</td>
<td>Policy in paying off checks (8 slides)</td>
</tr>
<tr>
<td>6. Cost estimation</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7. Function-oriented view</td>
<td>12</td>
<td>2</td>
<td>Setting of the digital watch (7 slides)</td>
</tr>
<tr>
<td>8. Data-oriented view</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9. Rule-oriented view</td>
<td>8</td>
<td></td>
<td>Policy in paying off checks (8 slides)</td>
</tr>
<tr>
<td>10. Structured analysis</td>
<td>19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11. State-oriented view</td>
<td>5</td>
<td></td>
<td>Setting of the digital watch (7 slides)</td>
</tr>
<tr>
<td>12. Scenario-oriented view</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13. Object-oriented analysis</td>
<td>13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14. Formal software specification and program verification</td>
<td>3</td>
<td></td>
<td>The tank (reservoir) (9 slides)</td>
</tr>
<tr>
<td>15. Overview of design activities</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16. Structured design</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17. Object-oriented design</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18. Implementation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19. Systematic testing</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20. Functional testing</td>
<td></td>
<td>- Building blocks (6 slides)</td>
<td>- Coverage test (23 slides)</td>
</tr>
<tr>
<td>21. Software metrics</td>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22. Maintenance</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23. Reverse engineering</td>
<td>12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24. Quality of software development process and its standardization</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25. Introduction to software ergonomics</td>
<td>??</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26. User manuals</td>
<td>??</td>
<td></td>
<td></td>
</tr>
<tr>
<td>27. Project management</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28. Configuration and version management</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assignment 1 – review of requirements specification document</td>
<td>√</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assignment 2 – cost estimation</td>
<td>√</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assignment 3 – review of the product model according to structured analysis</td>
<td>√</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assignment 4 – derive a use case and class diagram for a new software specification</td>
<td></td>
<td>An independent problem (req. spec.)</td>
<td></td>
</tr>
<tr>
<td>Assignment 5 – derive a formal specification for a new software subsystem</td>
<td></td>
<td>The queue</td>
<td></td>
</tr>
<tr>
<td>Assignment 6 – apply regression testing tool to a new small example program</td>
<td></td>
<td>An independent program (program source)</td>
<td></td>
</tr>
<tr>
<td>Assignment 7 – build a classification tree for one use case</td>
<td>√</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assignment 8 – apply some software metric tools to a new software</td>
<td></td>
<td></td>
<td>An independent program (program source)</td>
</tr>
</tbody>
</table>
2 The main case-study – Seminar Organization

2.1 Topic 5: Results of the “Analysis and Definition” phase

The main case study is for the first time mentioned and used in Topic 5: Results of the “Analysis and Definition” phase. It is used to show and describe the requirements documents for a software product.

**To do:**
Develop a preliminary requirements specification and requirements specification for a software product that should be:
- of similar size as the current one (based on the number of use cases, for example),
- business-oriented (with data), such that function point method can be applied for cost-estimation,
- of such complexity:
  - that use cases can be used to illustrate include, extend, and generalize relations,
  - that entity-relationship diagrams illustrating important notations can be created,
  - that decision tables illustrating important notations can be created,
  - that data-flow diagrams can be refined reasonably deeply,
  - that class diagram can illustrate all important aspects,
  - etc. (see sections for topics 7 – 13 in this text)

Two mentioned documents should:
- be based on use-cases,
- follow the structure and contents given in [3, 4] (e.g. graphical and textual representation of use cases, data, quality expectancy, etc.),

**Option:**
Preliminary requirements specification may not be produced. In that case however, one of the assignments should be changed (see Topic 6: Cost estimation for further details).

**Remark:**
It is also possible to use a different document structure (e.g. IEEE standard).

Excerpts from these two documents are used in the lecture (topic 5). Lecture also elaborates on how requirements can change over the time. Therefore,

**Option:**
Develop a previous version of the requirements. These two documents should follow the structure given in [5, 6] and should be not based on use-cases.

**Remark:**
Since it may be hard to produce these two documents, this activity can be omitted. In that case the topic should be changed by deleting corresponding slides.

Slides where the case study (i.e., the requirements specifications) are used are the following:

---

1 Please note that the slide title is a unique identifier of a slide inside the topic.
**General slides** describing only the summary of the product and giving an excerpt from the glossary (part of requirements specification).

**Example: customer’s request „Seminar Organization“**

A company for advanced training (on-the-job training) needs a computer-based system for the management of its lectures. In particular, it should be possible to administer seminars and participants, to issue invoices, to answer queries and to create statistics.

**Glossary**

- Defines notions to assure a *unified terminology*
- The glossary will be reused for the *user interface*, the *online help* and the *user manual*.
- **Examples:**
  - Seminar organization: 12 notions
  - XCTL (control program in physics): 110 notions

**Example of a glossary (excerpt)**

**Seminar organization**

<table>
<thead>
<tr>
<th>Version</th>
<th>Author</th>
<th>Date</th>
<th>Status</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>Balzert</td>
<td>31.07.2000</td>
<td>accepted</td>
<td></td>
</tr>
</tbody>
</table>

**Client**

Associate of a company or a private person, who is interested in services, or have booked and participated the seminar.

**Client manager**

Responsible for communication with clients and companies, together with booking and information providing.

**Company**

Associate of a company (costumer) who is responsible for education and further education of company employees and who is informed about services or who sends associates or public presentations, or who books for closed presentations.

**What should be more precisely specified in this example requirements specification before starting the product development?**

**Slides describing documents** that show characteristic parts of both documents.

**Example of a requirements specification (excerpt)**

**Seminar organization**

<table>
<thead>
<tr>
<th>Version</th>
<th>Author</th>
<th>Date</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Balzert</td>
<td>03/91</td>
<td>accepted</td>
</tr>
<tr>
<td>2.2</td>
<td>Balzert</td>
<td>10/91</td>
<td>F115/ added</td>
</tr>
<tr>
<td>2.3</td>
<td>Balzert</td>
<td>10/95</td>
<td>F15/125/ F165/ removal.</td>
</tr>
<tr>
<td>3.0</td>
<td>Balzert</td>
<td>31.06.60</td>
<td>Extension on the Web</td>
</tr>
</tbody>
</table>

**1 Goals**

The seminars presented by "Teachware" company should be supported by computers.

1.1 Compulsory criteria

- managing seminars
- managing presentations
- managing clients (participants/interested parties)
- managing client companies
- managing lectures
- queries like
  - Who will the next 2 seminars take place?
  - Which associates participated the seminar X?

1.2 Optional criteria

- all compulsory functions (the compulsory criteria) should be accessible through Internet (Web browser)
- hotel and contact person management
- statistic evaluation
- data security support

1.3 Exclusion criteria

- No accounting (book keeping) integrated (the accounting has a copy of invoice and keeps track of payment and notifies of the paying party)
2 Product Usage
The product is used by client-, company-, lecturer-, seminar- and presentation management of "Teachware" company. Besides that, various queries can be answered.

2.1 Application area
Salesman/administrative application area.

2.2 Target Groups
Associates of "Teachware" company should be divided into: client manager, seminar manager, presentation customer.
"Teachware"'s clients and companies can get the information about seminars and presentations on the Internet. They can book using Internet, as well.

4 Product functions

4.1 Use cases
F10 (PF10)
Use case: informing:from question to information
Goal: client gets required information or the information material is sent to him/her.
Category: primary
Preconditions:
Post condition success: client gets required information
Post condition failure: the required information can not be issued
Actors: client manager, client, company
Triggers: client writes (letter, fax, e-mail) or calls.
Description:
1. client data retrieval
2. information issue
Extension:
1. A client data actualization
2. A production of address label (for sending info-material)
Alternative:
1. An inclusion of a new client

4.2 Lists
F70 (PF70)
Participant list: a) per seminar with following data: seminar title, starting date, finishing date, presentation place, lecturers. b) per participant: first name, family name, company, town.
F80 (PF80)
Participant certificate: for every seminar participant with following data: address, title, first name, family name, starting date, finishing date, seminar title, place, overview, conductor
F90 (PF90)
Queries like the following should be allowed:
When the next X seminar will be held?
Which associates of company Y participated in seminar X?

5 Product Data

5.1 Client Data
D10 (PD10) Client data (max. 50,000)
Client number, name, address, communication data, date of birth, function, exchange, short information, notice, info material, client since
D20 (PD20) Company data (max. 10,000), when a client is an associate of a company:
Company's short name, company name, address, communication data, contact person, section, date of birth, function of contact person, short information, notice, exchange, client since
D21 If a company is in a paying delay, then the following data should be saved:
Date of still unpaid invoice, as well as amount
5.2 Seminar Data

D50 (PD500): Seminar data (max. 100,000):
- Seminar number, duration (in days), from, to, daily period split begining, daily period split end, beginning of the first day, end of the last day, seminar place (hotel, company, address, room), cooperation partner, public (yes/no), net price, cancel fee, min. participant rate, max. participant rate, actual participant, carried out, net price.

D40 (PD40): Seminar type data (max. 10,000):
- Short title of seminar, seminar title, purpose, method, overview, daily procedure, duration, records, target group, requirements, fee without tax, min. participant rate, max. participant rate.

D56 (PD56): Lecturer data (max. 5,000):
- Lecturer number, name, address, communication data, date of birth, biography, daily attendance, short information, notice, lecturer since.

D60 If a lecturer conducts a seminar, the information should be saved.

5.3 Booking Data

6. Performance concerning time and amount of data.

8 User Interface

U10 Standard Windows-oriented environment.

U20 The web-browser handling is simplified. The available functions are executed in side-ways frames. In main frames are presented the lists and register masks.

U30 Service interfaces are designed for mouse.

U40 ISO 9241-10: 1996 (Ergonomic requirements for office work with screen machines, part 10: dialog design fundamentals) to be taken into account.

U50 To distinguish the following roles:

- Role
  - Client manager: F10, F20, F31, F60
  - Seminar manager: F22, F23, F40, F50, F60, F90
  - Seminar custodian: F30, F70, F80
  - Lecturer: F70, F80 (for some presentations only through Internet)
  - Client, company: F10, F30, F21 (only through Internet)

12 Structure of Project Parts

There are three parts planned. First version covers core functionality without Internet, second one covers core functionality expanded with some Internet functionality like booking and booking the company's internal seminar. The third version supports hotel and terminal management.

SemOrg V1.0 (Core)

- F10 Inquiry: inquiry from question to information. (without Internet)
- F20 Booking: from registration to booking. (without Internet)
- F30 Preparing seminar: from registration to booking. (without Internet)
- F40 Acquiring lecturer: from choosing to engaging. (without Internet)
- F50 Planning presentation: from scheduling to reservation. (without Internet)
- F70 Participant list: (without Internet)
- F80 Participant certificate: (without Internet)
- F90 Canceling: from canceling to cancel notification. (without Internet)
- F21 Checking out: from canceling to cancel notification. (without Internet)
- F22 Booking company: from registering to booking a company's internal presentation. (without Internet)

SemOrg V2.0

- F20 Booking: from registration to booking. (with Internet)
- F30 Preparing seminar: from registration to booking. (with Internet)
- F40 Designing seminar: from idea to a new seminar. (with Internet)
- F50 Acquiring lecturer: from choosing to engaging. (with Internet)
- F60 Planning presentation: from scheduling to reservation. (with Internet)
- F70 Participant list: (without Internet)
- F80 Participant certificate: (without Internet)
- F90 Canceling: from canceling to cancel notification. (with Internet)
- F21 Checking out: from canceling to cancel notification. (with Internet)
- F22 Booking company: from registering to booking a company's internal presentation. (with Internet)

13 Supplements

According experience, 5% of all clients are in paying delay.

Slides comparing two versions of requirements (optional) that summarize differences between the previous and the current requirement specifications.
2.2 Topic 6: Cost Estimation

Documents produced in the previous step are used to calculate cost estimation that will be partly shown during this topic.

To do:
Develop a cost estimation calculation using a function point method, based on preliminary requirements specification produced in previous step.

Option:
If the preliminary requirements specification has not been produced in the previous step, then cost estimation must be shown on requirements specification! Since this was intended as a student assignment (see later), in this case another student assignment (i.e. example) must be devised.

**Remark:**
It is also possible to use another cost estimation method (e.g. COCOMO) but it should be done only as an additional method. Function point (at this time) has the priority.

Slides where the cost estimation calculation is used are the following\(^2\):

*Introductory slides* used to support the introduction of FP method.

---

\(^2\) Please note that the slide title is a unique identifier of a slide inside the topic.
**Detailed slides** showing detailed parts of calculation.

---

**Example: Preliminary Requirements Specification „Seminar organization“ V 2.3 (1)**

- **/PF 10/ Adding new, changing and deleting customers’ data (participants, prospects).**

  - There are three separate inputs (Adding, Changing, Deleting).

  - Adding a new client is certainly most extensive. There are probably more than 10 data elements to be gathered, a logic input correctness check is needed (consistency check: zip-code/place), a writing access to the data base is needed, user guidance is expected to be high (automatic positioning of cursor, field centered editing). Adding a new client is a complex input.

---

**Example: Requirements Specification „Seminar organization“ V 2.3 (2)**

- **/PF 10/ contd.**

  - During a change of client’s data the data base is read from and written on. User guidance is needed to be usual; the number of changed data elements may vary from small to high. Therefore classifying this input as middle seems sufficient.

  - Deleting a client’s entry demands logical checks and a data base access on seminar bookings /LF50/.

  - Deleting is therefore also classified as being of middle complexity.

  - Result: 1 complex input, 2 middle inputs.

---

**Example: Requirements Specification „Seminar organization“ V 2.3 (3)**

- **/PF 20/ Information of customers (registration affirmation, checkout affirmation, change information, invoice, advertising)**

  - There are five separate outputs.

  - Because there are no specifics in the requirements specification available and most of these outputs are combinations of a few data elements with some data and standard texts, they are classified as being of middle complexity.

  - Result: 5 middle outputs.
Example: Requirements Specification „Seminar organization“ V 2.3 (4)

**PF30/**
As in „PF10/“ but respectively for seminar events and seminar types.
Result: 2 complex and 4 middle inputs

**PF40/**
As in „PF10/“
Result: 1 complex and 2 middle inputs

**PF50/**
To book a seminar it is only necessary to link the customer with the corresponding seminar event. So there are only a few data elements involved, however a logical check with data base access is needed. These 3 inputs are classified as being of middle complexity.
Result: 3 middle inputs

Function-Points am Beispiel: Lastenheft „Seminarorganisation“ V 2.3 (6)

**LF60/**
Queries similar to the following should be answered:
When will the next seminar X take place?
Which company Y’s associates participated the seminar X?

These are queries with end user languages. They do not count.

Example: Requirements Specification „Seminar organization“ V 2.3 (5)

**SPFC**
An invoice has to contain data on the customer, the seminar event and the seminar type. This requires some data base accesses. The output will probably contain more than 10 data elements. This leads to a complex output.
Result: 1 complex output

**PF70/**
As in „LF60/“ these are three complex outputs.
Result: 3 complex outputs

Example: Requirements Specification „Seminar organization“ V 2.3 (7)

Product data

**PD10/**
This should be one simple data stock (1 key, number of different data elements < 20).
Result: 1 simple data stock

**PD20/**
As in „PD10/“ this is one simple data stock.
Result: 1 simple data stock

**PD30/**
As in „PD10/“ respectively for seminar event, seminar type and lecturers.
Result: 3 simple data stocks

**PD40/**
As in „PD10/“.
Result: 1 simple data stock

Un-weighted Function Points

| Input data: | 11 x middle (4) = 44 |
| Output data: | 4 x complex (6) = 24 |
| Data: | 5 x middle (5) = 25 |
| Function Points sum (E1) | 42 |

Influencing factors

The influencing factors are considered as follows:
1. Integration with other applications (0-5):
2. Decentralized data processing (0-5):
3. Transaction rate (0-5): because of „PF10/“, efficient DB access
4. Processing logic
   a. Arithmetic operations (0-10): more complex algorithms
   b. Control procedures (0-5):
   c. Exception handling (0-10): special cases
   d. Logic (0-5):
5. Reusability (0-5):
6. Data stock conversions (0-5):
7. Adaptability (0-5):
Sum of the seven influences: E2:
2.3 Topic 7: Basic concepts of the functional view

Requirements for a software product should now be analyzed according to several methodologies/views. First we take into consideration a functional view and should illustrate function tree, data flow diagram, and use case diagram on the requirements specification of the case study.

To do:
- Develop a full data-flow diagram of requirements – it will be needed also later.
- Develop a function tree of main functions of requirements (function tree is implicitly contained in a data-flow diagram, so it just have to be recognized).

Option:
In fact not the full data flow diagram is needed, but at least three subsystems have to be fully developed.

Remark:
If requirements specifications are developed as requested previously, then there is no special activity related to use cases in this lecture – we shall just use some excerpts from the already produced document.
This topic also elaborate on the difference between functions and use cases near the end, using the example from two versions of requirement specification: the old one without use cases, and the new one with use cases.

**Remark:**
If the previous version of requirements has not been produced earlier, then this elaboration must be illustrated differently in this topic, i.e., changes will be necessary.

*Slides on function tree.*

*Slides on data-flow diagram.*

*Slides on use cases.*
2.4 Topic 8: Basic concepts of data oriented view

Data dictionary and entity-relationship model should be illustrated with the case study.

To do:
- Design a user-interface form for which a relatively complex data dictionary can be created (see slides). Such data dictionary should illustrate most important data dictionary notations.
- Develop several entity-relationship diagrams from case study illustrating important entity-relationship notations (see slides).

Slides related to case study can be grouped in two groups:

Slides related to data dictionary.
2.5 Topic 9: *Basic concepts of the rule oriented view*

Decision tables should be illustrated with the case study.

**To do:**
- Develop an example suitable for description with decision tables (checks, rules, sequence of activities that can be done under certain conditions, …) (see slides).

All slides related to case study belong only to one group:

*Slides illustrating decision tables.*
2.6 Topic 10: Structured analysis

This topic is essentially driven by the data-flow diagrams developed for the case study. They are gradually refined, and finished with mini-specifications and data dictionaries. This topic also presents once more an already developed function tree that is implicitly contained in diagrams.

To do:
- Develop an example for mini-specification for at least one illustrative data flow diagram.
- Develop an example for a data dictionary that will be used with refinement of data flow diagrams.

All slides related to case study belong to three groups:

- Slides illustrating refinement of data flow diagrams.
Slides illustrating mini-specifications (and function tree).
Slides illustrating data dictionary and its use with data-flow diagrams.

DD Entries: „Seminar Organization“ (1)

- With the transition from context diagram to DFD 0 no data flows have been refined
  - New data flows between storages and processes were inserted
  - In DFD 4 data flows were refined
    - This has to be recorded in the data dictionary.

  request data = personal data + (company data)
  booking data = registration data + check out data

- If data flow names in different DFDs are identical, then they are concerned with the same data flows.
2.7 Topic 11: Basic concepts of state oriented view

The topic uses case study to illustrate notational elements of automata and activity diagram. Also, two slides of with excerpts from two CASE tools are given.

To do:
- Develop an example for an object life cycle that can illustrate important notational aspects (see slides).
- Develop an example for an activity diagram that can illustrate important notational aspects (see slide).

**Remark.**
Additional slides can be also produced.

All slides related to case study belong to three groups:

**Slides illustrating object life cycle.**

**Slide illustrating activity diagram.**

**Slides illustrating CASE tools.**
2.8 Topic 12: Basic concepts of scenario-based view

The topic uses case study to illustrate principles of sending messages and notational elements of collaboration diagram and sequence diagram.

**To do:**
- Develop an example that can be used to illustrate the principle of sending messages in OO world. (see slides).
- Develop an example for a collaboration diagram that can illustrate important notational aspects (see slide).
- Develop an example for a sequence diagram that can illustrate important notational aspects (see slide).

**Remark.**
Additional slides can be also produced.

All slides related to case study belong to two groups:

*Slides illustrating sending of messages.*
2.9 Topic 13: Object-oriented analysis

This topic uses case study to introduce all notational aspects of class diagrams and to elaborate on possible other in designing classes. Topic also uses already presented slides form Topic 10: Structured Analysis.

To do:
- Develop a full class diagram for a case study, in such a way that elaboration of other possibilities (choice of classes, etc.) can be done.

All slides related to case study belong to three groups:

Slides comparing OOA with SA.
Slides introducing notations and problems (other possibilities) of class diagrams.
2.10 Topic 14: Formal specifications

Parts of already developed documents/slides are used here in order to clarify the need for formal specifications. No special effort is needed here, except to use and slightly adapt already existing slides.
2.11 Topic 16: **Structured design**

This topic uses case study to introduce basic principles of structured design. It takes one data-flow diagram and proceeds in designing the software for it.

**To do:**
- Develop a structured design for at least one data-flow diagram from structured analysis..

All slides related to case study belong to one group:

*Slides introducing notations and principles of structured design.*
2.12 Topic 16: **Object-oriented design**

This topic uses just one slide describing a case study – the slide that elaborates on how class libraries can be used. However, this slide can remain even if the case study is changed, because it is general enough.
2.13 Topics 25 and 26: *Introduction to software ergonomics and User manuals*

These two topics are not yet available in English. Nevertheless:

To do:
- Implement a case study or provide a design of user interface.
- Write parts of user manual.

2.14 Summary

Activities in building a new case study are summarized as follows (this is only an approximation – for full description, you should nevertheless read the whole text). Activities are ordered by importance:

- Find a problem of reasonably large size an complexity (for example from textbooks, real projects or educational projects)
- Develop requirements specification
- Develop a full class diagram as the basis of object-oriented analysis
- Develop accompanying diagrams for the dynamic view of object-oriented analysis: state automata (object life cycle), activity diagrams, collaboration diagram, sequence diagram
- Develop a data-flow diagram for a significant part of requirements
- Perform the structured analysis of the system: develop a hierarchy of data flow diagrams for a significant part of the requirements
- Do a cost estimation
- Implement the case study
- Write parts of user manual

To replace the existing case study with the new one, one should replace about 120 slides in the lecture.
3 The supporting case-study – XCTL

XCTL is a realistic program used mainly in the lecture to illustrate a process of reverse engineering. To find out its basic characteristics a software measurements has been applied and also presented in the lecture.

In the same way another case study could replace XCTL or illustrate other important aspects of software development.

3.1 Topic 5: Results of the “Analysis and Definition” phase

The supporting case study is for the first time mentioned in Topic 5: Results of the “Analysis and Definition” phase. It is used only in one slide to show how many notions are there in the glossary.

To do:
Develop at least glossary and use case diagram (as part of requirements specification) for a software product, that should be:
- of similar size as the current one (based on the number of use cases, for example),
- possibly already existing and of unknown structure (such that it is suitable for finding out the structure ☺)
- of such complexity:
  - that various software metrics methods can be applied

Option:
Develop a full requirements specification.

3.2 Topic 7: Function-oriented view

This topic uses the supporting case study to show its use cases.
3.3 Topic 21: Software metrics

This topic uses the supporting case study to illustrate the application of several software metrics methods (see slides).

To do:
- Develop several characteristic measurements that can be used to illustrate major measurement techniques
Measurement results (graphical summary and evaluation)

Cumulative Histogram of most important complexity metrics

How many % of functions go over metric values?

Special values are put as limits on the x coordinate. Considering each of measures between 7 and 9 %, (65-85) of calls have bad values concerning fixed limits.

Manual inspection:
High metric values really a problem (1)?

Manual inspection:
High metric values really a problem (2)?

Scatterplot
(relation between v(G) and ev(G), reliability and maintainability)
3.4 Topic 23: Reverse engineering

This topic uses supporting case study to show a realistic process of a reverse engineering.

To do:
- Develop (or simulate) a similar process to be shown to students.

Slides are used in two ways in the lecture.
*Slides illustrating wrapping and giving overview of used tools.*
Slides illustrating the whole process of reverse engineering, incl. history.

Emails with a physicist
First inquiry by the physicist (25.06.98) (1)

Dear Mr. Bothe,
I turn to you with an appeal for an advice. We are an experimental working group at an institute for physics. In our laboratory there are approximately 10 computers, which steer Roentgen-diffraction meter. Some years ago an engineer wrote a program (C++) for Windows 3.1, which is used to control nearly all these devices. This program communicates with different Motor and detector controlling cards. Unfortunately, the hardware control is to a large extent hidden in a very large DLL, and the program code altogether is not very clear.

Emails with a physicist
First inquiry by the physicist (25.06.98) (2)

My questions are now the following:
- Could you or one of your colleagues advise us, as how could we best come to a tidy software basis for the future, which we ourselves can maintain at bearable expenditure.
- An ideal version (as we see it): perhaps the problem of control and adjustment is interesting enough for you also, in order to assign it as topic of a diploma thesis.

Yours sincerely,
Rolf Koehler

Questions

Should the order be accepted?

Which problems should be clarified first?
Emails with a physicist
Answers to the further inquiries (01.07.98) (1)

Dear Mr. Belott...

1. Is the engineer, who developed the program, still there?
   Unfortunately no, but he could be reached by email or telephone if necessary.

2. How well is the program documented and commented?
   A formal overview, who studied in the meantime technical informatics at a
   higher craft school, was so nice to look after the program. According to his
   statement the program is sufficiently commented. The interfaces of DLLs
   are documented. The program is already developed using object-oriented
   approach, but it is not structured well. In particular, the access to the
   hardware is bundled together with many other functions in a very large DLL
   (also with some elements of the user interface).

3. How large is the program (in lines of C++ source code)?
   The source code consists of 30-40 files with the average size of 20k bytes
   each.

4. Is it grounded on some special design method, e.g. with the help of the object-
   oriented approach?
   See above. According to statement of the mentioned formal overview only
   object-oriented elements were later inserted.

Emails with a physicist
Answers to the further inquiries (01.07.98) (2)

5. Do you expect the existing program to be restructured (reorganized) or a
   new one developed?
   It will be probably inevitable to restructure the program completely. It
   cannot say, to what extend parts of it are further usable. At least one
   must recognize for a part of the hardware, how it is stimulated. Even
   this relief is not probably entirely applicable, since it functions
   completely by polling, and possibly use of a hardware interrupts in
   individual places is more meaningful. The programs are not time-
   critical. So far Windows 3.1 is installed on all measuring computers
   concerned. Upgrading computers so that Windows NT can be used is
   pratically ruled out for now. Since you can hardly still assign a
   programming for Windows 3.1 as a task, we could imagine the use of
   Linux, since it has smaller hardware requirements.

Emails with a physicist
Answers to the further inquiries (01.07.98) (3)

6. Which concrete problem do you have with the existing program: error detection,
   extension, porting on other computers...?
   There are the following problem fields:
   1. Instability: partly with certain actions, partly completely, coincidentally, the program fails.
   2. Error: Even essential things appear: e.g. a responding of end position switches is not
      always reliably recognized, and has already led to conditions that endangered
      the mechanics. However, we cannot completely exclude that this has to do also with
      the used control cards.
   3. Representation of measurement results: Among other things a linear position sensitive
      detector is used with multi-channel analyzer. The data are stored in two-
      dimensional field. For the evaluation of the measurement, a graphical representation
      is necessary after a coordinate transformation (view, defined lines). That is at present
      realized using a pseudo-colored bitmap. In principle, it is sufficient, but the
      transformation does not turn out correctly, and the displayed image is not satisfying
      (no solid bitmap, color coding is unfortunatable and not flexible enough).
   4. Extension: The used motor control cards are no more produced. It would be necessary
      to take in successors into account. Further we must soon consider a new type of
      detector, which captures data in 1-or perhaps even two-dimensions. We do not see a
      possibility to switch to another type of computer, but to use the existing
      PCs further on.

Emails with a physicist
Answers to the further inquiries (01.07.98) (4)

7. Which C++ and/or which C++ compiler was the basis for development?
   Borland C++, last version 4.52.

8. How urgent is the work?
   Task provided within 12 months is adequate.

9. Who will take care of the program in the future, and with which goal (e.g. larger
   extensions planned)?
   That is a difficult question. Naturally, it would be outstanding, if we could get
   external support to it. Since that is not guaranteed, it would be nice, if those
   parts, which concern the pain flow control, are sufficiently isolated to be
   affordable without deeper knowledge of the entire program. Then our
   graduate students and/or our engineer (without training in programming)
   could adapt the program to changing measuring tasks. An integration of new
   hardware in the same way is surely not possible. However, we will probably
   have 2 new detectors, whose integration could flow if necessary into the
   task in the next months. Likewise we would like to buy a new motor control
   card, so that can be likewise directly considered. I would be very grateful, if
   you could actually support us concerning the program.

Yours sincerely Rolf Koehler

RTK Project: found situation (1)

- Program sources:
  - 46 files, sizes of 01 - 33770 LOC (cpp Files)
  - A few usable comments
  - Files: not always problem adequately (logically) arranged partly bad layout (formatting)
  - Identifiers formed uniformly in the same style (well?)
  - Inadequacies in details: e.g. switch with a single case...
  - Dead Code: 90 functions (implemented - not called)
  7 classes / 4 structures (declared - not used)

RTK Project: found situation (2)

- SW architecture (paradigm):
  - Mixture of procedural and object-oriented parts (C/C++)
  - object-oriented approach well (user interface among other things)

- User interface not ergonomic
  - Arrangement of the elements of a window

- None of the following documents:
  - Product requirement specifications, SW architecture, user manual, test documentation, program documentation

- Programmer no more in place
  - Physicist (esp. one) co-operation friendly
Multi-term Project: Organization

Multi-term Project: Statistics

Decomposition into tasks: Starting situation

27,000 LOC:

45 Files with:
- includes: 158
- macros: 639
- functions: 30
- variables: 562
- types: 74
- enums: 25
- classes: 58
- instance vars: 578
- instance methods: 457

Use Case Diagram
3.5 Summary

Activities in building a new supporting case study should be summarized as follows (this is only an approximation – for full description, you should nevertheless read the whole text).

To replace the existing case study with the new one, one should replace about 33 slides in the lecture.

A supporting case study can be developed either to replace XCTL (mainly in topics on software metrics and reverse engineering) or to illustrate other important aspects of software development.
4 Possible further extensions

This section lists possibilities to further include the main case study into the lecture’s topics.

<table>
<thead>
<tr>
<th>Topics</th>
<th>Currently</th>
<th>Possibilities to include also in</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. What is software engineering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Quality criteria for software products</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. Software process models</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Basic concepts for software development documents</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5. Results of the “analysis and definition” phase</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>6. Cost estimation</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>7. Function-oriented view</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>8. Data-oriented view</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>9. Rule-oriented view</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>10. Structured analysis</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>11. State-oriented view</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>12. Scenario-oriented view</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>13. Object-oriented analysis</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>14. Formal software specification and program verification</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15. Overview of design activities</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16. Structured design</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>17. Object-oriented design</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>18. Implementation</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>19. Systematic testing</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>20. Functional testing</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>21. Software metrics</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>22. Maintenance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23. Reverse engineering</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24. Quality of software development process and its standardization</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25. Introduction to software ergonomics</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>26. User manuals</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>27. Project management</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28. Configuration and version management</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assignment 1 – review of requirements specification document</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>Assignment 2 – cost estimation</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>Assignment 3 – review of the product model according to structured analysis</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>Assignment 4 – derive a use case and class diagram for a new software specification</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assignment 5 – derive a formal specification for a new software subsystem</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assignment 6 – apply regression testing tool to a new small example program</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assignment 7 – build a classification tree for one use case</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>Assignment 8 – apply some software metric tools to a new software</td>
<td>√</td>
<td></td>
</tr>
</tbody>
</table>
References