Seminar in Empirical Software Engineering

Session 1
Agenda for today’s session

- Overview of the seminar
- Introduction to Empirical Software Engineering
- Seminar format
- Seminar work topic
Overview of the seminar: Focus and purpose

• The seminar focuses on contemporary topics in software engineering research and practice
  • Topics that are relevant in today’s software development practice
  • Topics which are covered in recent scientific research
• The purpose of the seminar is to
  • Practice written scientific communication
    – Reading scientific text, finding material, understanding paper structure
    – Writing scientific text, the process of writing and improving text
    – Analysing and critiquing scientific text
  • Practice oral scientific communication
    – Presenting ideas and results verbally and discussing them
• Learn something about empirical software engineering research!
## Overview of the seminar: Sessions

<table>
<thead>
<tr>
<th>Session</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>September 7</td>
<td>Opening, introduction, potential topics</td>
</tr>
<tr>
<td>September 14</td>
<td>Introduction to scientific writing in the seminar, conference submission and</td>
</tr>
<tr>
<td></td>
<td>review process, topic assignment</td>
</tr>
<tr>
<td>September 21</td>
<td>Preparing a presentation, doing literature reviews</td>
</tr>
<tr>
<td>October 12</td>
<td>Feedback session</td>
</tr>
<tr>
<td>November 30</td>
<td>Conference Day I (presentations)</td>
</tr>
<tr>
<td>December 7</td>
<td>Conference Day II (presentations)</td>
</tr>
</tbody>
</table>
# Overview of the seminar: Activities and deadlines

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>September 28</td>
<td>Deadline: <strong>Extended abstract</strong></td>
</tr>
<tr>
<td>October 19</td>
<td>Deadline: Paper bidding</td>
</tr>
<tr>
<td>November 9</td>
<td>Deadline: <strong>Full paper</strong></td>
</tr>
<tr>
<td>November 12</td>
<td>Review invitation</td>
</tr>
<tr>
<td>November 23</td>
<td>Deadline: <strong>Review</strong></td>
</tr>
<tr>
<td>November 30</td>
<td><strong>Presentation</strong> (Conference Day I)</td>
</tr>
<tr>
<td>December 7</td>
<td><strong>Presentation</strong> (Conference Day II)</td>
</tr>
<tr>
<td>December 21</td>
<td>Deadline: Paper revision (optional)</td>
</tr>
</tbody>
</table>

Note: Deadlines are at 12 noon. The deadlines are strict!
Overview of the seminar: Grading

• The grade is based on points accumulated during the seminar
• Points are divided between
  • The seminar paper (40%)
  • Reviews given to other students and activity in the seminar sessions (30%)
  • The seminar presentation (30%)
• You must pass all three areas
  • The criteria for Bachelor’s and Master’s theses apply to the paper
  • Criteria for reviews will be covered in another session
  • Seminar activity means attending sessions and turning in assignments on time
  • Criteria for presentations will be covered in another session
Introduction to Empirical Software Engineering
Introduction to Empirical Software Engineering

• Software engineering focuses on understanding, controlling, managing, and improving software products and processes based on engineering principles (Boehm, Brown & Lipow, 1976; Basili & Weiss, 1984; Basili & Rombach 1988)

• A major goal is the delivery of high-quality software

• Software engineering research asks questions about the methods used to construct and evaluate software

• It is also concerned with the context in which those methods are applied

• It is often multidisciplinary, linking to various other fields including natural, social, and behavioural sciences and humanities
Introduction to Empirical Software Engineering

• Empirical research seeks to gain **knowledge** by using **empirical evidence**
• Empirical evidence is acquired through **observation** and/or **experimentation**
• Empirical evidence justifies a belief in the truth or falsity of a claim
• **Knowledge** means having a true belief based on empirical evidence
• In contrast, a rationalist view is that reflection or analytical thinking is considered evidence
Introduction to Empirical Software Engineering

Belief: Pair programming works better than solo programming.

<table>
<thead>
<tr>
<th>Empirical</th>
<th>Gut feeling / opinion</th>
</tr>
</thead>
<tbody>
<tr>
<td>+  The evidence shows that</td>
<td>It has to work better. At least I like it. Two persons get more done. Two persons create better quality.</td>
</tr>
<tr>
<td>• for low-complexity programming tasks, pair programming is faster than solo programming, and</td>
<td></td>
</tr>
<tr>
<td>• for high-complexity tasks, pair programming yields higher quality code solutions.</td>
<td></td>
</tr>
<tr>
<td>-  The evidence shows that</td>
<td></td>
</tr>
<tr>
<td>• the reduced completion time for the simpler tasks comes at a price of noticeably lower quality, and</td>
<td>It doesn’t work better. I wouldn’t want someone peeking while I code. You just start to cut corners. It just takes longer to explain everything.</td>
</tr>
<tr>
<td>• the higher quality for complex tasks comes at the price of considerably greater effort.</td>
<td></td>
</tr>
</tbody>
</table>

(Hannay et al., 2009)

• Empirical research often gives complex answers because reality is complex.
• Gut feeling can be important because it can influence what happens, and it can be true. It is still not knowledge!
Quick pair discussion

- What questions do you know (or can think of) that
  - Have an answer in empirical software engineering research?
  - Could have an answer in empirical software engineering research (but maybe you don’t know if there is any research on it)?
Examples of research objects

- Organisations
  - e.g. a specific division in a company

- Software Processes
  - e.g. test-driven development

- Projects
  - e.g. requirements documents, software

- Tools
  - e.g. Eclipse IDE

- Products
  - e.g. professional software developers

- People

Empirical Software Engineering Research

- e.g. The Apotti health care system project
## Examples of research strategies

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Quantitative (data expressed as numbers)</th>
<th>Qualitative (data expressed as words or pictures)</th>
<th>Environment</th>
<th>Logic</th>
<th>Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controlled experiment</td>
<td>X (measurements)</td>
<td></td>
<td>Laboratory</td>
<td>Manipulate one or more variables, keep others at fixed levels</td>
<td>Measure change in dependent variable</td>
</tr>
<tr>
<td>Natural experiment</td>
<td>X (measurements)</td>
<td></td>
<td>Field</td>
<td>Natural causes have resulted in changes in one or more variable</td>
<td>Measure (change in) dependent variable</td>
</tr>
<tr>
<td>Case study</td>
<td>X (measurements)</td>
<td>X (interviews, documents, etc.)</td>
<td>Field</td>
<td>Studying multiple cases can reveal generalisable patterns</td>
<td>Observe events and patterns (Can also use measurement)</td>
</tr>
<tr>
<td>Survey</td>
<td>X (numeric questionnaire data)</td>
<td>X (interviews, open questionnaire data)</td>
<td>Field</td>
<td>Responses reveal information about the sample</td>
<td>Statistical analysis (closed questions) Qualitative analysis (open questions)</td>
</tr>
<tr>
<td>Meta-study</td>
<td>X (data extracted from articles)</td>
<td>X (analysis of article text)</td>
<td></td>
<td>Primary studies can be used to answer research questions</td>
<td>Data extracted from primary studies is used to answer questions</td>
</tr>
</tbody>
</table>

http://dx.doi.org/10.1145/2884781.2884837

• Identify research object, method, and results

ABSTRACT Job Rotation is an organizational practice in which individuals are frequently moved from a role in projects to another in the same organization. Little in the literature has discussed the effects of job rotation on software engineering teams. The goal of this study is to investigate the effects of job rotation on software engineering teams. The findings of this study are significant not only because software engineers have a direct impact on the effectiveness of software projects but also because organizations are increasingly engaging in job rotation. In this study, we developed the practice of job rotation in a software organization guided by the following research questions:

1. How does the practice of job rotation affect software engineers at work?

We are inspired to present this project and results, as described by Woods [2] in which individuals were surveyed during the development of a software project, to work in another project’s role in the same or in a different role. To that end, we may research questions, we might be able to quantitatively analyze job rotation in software engineering teams in order to understand how job rotation, both positive and negative, is experienced by software engineers. We are interested to compare the objective of collecting data about software engineers’ experiences with the practice of job rotation in this company. This study included qualitative research about the impact of job rotation on the work of software engineers and the development of a theory of job rotation in software engineering.

The remainder of this article is organized as follows. In Section 2, we present the background and related work. In Section 3, we describe the research method and key findings of the study, including specific implications for the practice of job rotation. Finally, in Section 4, we conclude and discuss future work.

2. BACKGROUND

This section presents the background about job rotation that supports this study, and as related work.
Seminar format
Seminar format

• Conference-style working
  • Write a paper
  • Submit it to the conference
  • Peer review: write reviews
  • Conference presentation
  • (Optional: improve the paper)
• Sessions to support the seminar work
  • Three sessions introducing the topic area, scientific writing, conference format, giving presentations, doing literature reviews
Seminar work topic
Seminar work topic: Choosing a topic

• Choose a topic that interests you
• Look for some options, but do not spend too long choosing
  • You cannot go through everything 😊
• A narrow topic is better than a wide topic
• Some options:
  • Choose a topic based on literature that you find
  • Choose a topic based on experiences or practices in software companies or other organisations (still have to find relevant literature)
  • Choose from suggested topics
Choosing a topic based on literature

- Try searching in literature databases
- Use keywords that describe your interest
- Identify a few (1-3) key articles to use as a starting point
- Access databases through E-library portal: http://www.nelliportaali.fi/

(remember to log in to access all available material)
Choosing a topic based on experiences or practice

- Write a short description of what you have seen in a software development company or organisation
- Identify the major themes in your description: what is the problem or challenge?
- Do literature searches on the major themes
  - Is there any existing work on these themes?
  - What methods have been used to investigate them?
  - What results exist?
- Pick one theme to write about based on related literature
  OR
- Pick one method and apply it in your own scenario
  - Your report will document a study of your own
  - You still have to include related work and method literature
Choose from suggested topics

• In order to narrow your search, we suggest topics from the following sources:
  • The ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)
  • Information and Software Technology journal (recent issues): http://www.journals.elsevier.com/information-and-software-technology
  • Empirical Software Engineering journal
    – Online first articles (articles that will appear in a future issue): http://link.springer.com/journal/10664/onlineFirst/1
  • Master’s thesis topic list of the Empirical Software Engineering research group: https://www.cs.helsinki.fi/group/ese/topics_v01.html
Seminar work topic: Criteria for topics

- There must be a **sufficient number of papers** on the topic
  - Usually 5-10 articles of which a majority are full papers
- The topic must be **related to software engineering**
  - Typically this means there are publications in software engineering forums
- There must be **recent research** on the topic
  - For example, at least one paper from 2013 or later
- These are guidelines – ultimately, we decide on a case-by-case basis
Exercise: Topic brainstorming/development

• Split into groups with 4-5 persons
1. Individually, write down any number of topic ideas
   • These can be very preliminary at this point
2. Each person presents one idea in turn and the group helps to analyse it
   • Ask clarifying questions: what does the topic mean?
   • Search for literature: is there any work on the topic? What methods have been used in the literature? What results have been found?
   • Refine the terms: do articles use different words for the same thing?
   • Are there any related topics of interest mentioned in the articles?
   • Is this a feasible topic for a seminar work?
   • Record your progress by taking notes on the topics
   • Repeat the process until you have analysed all the topics
   • You may come up with new potential topics while you go
Exercise: topic brainstorming/development

- Summary from each group
- Discussion about topics and how to choose one
Next session
September 14

• Introduction to scientific writing in the seminar
  • What do we expect from the seminar paper?
  • How can you reach the level you are aiming for?
• Conference submission and review process
  • What are the steps in the process?
  • How does the submission system work?
  • What do we expect from reviews?
• Topic assignment
  • Homework: choose a topic
References


