Product Line Engineering Lecture – Introduction (1)

Dr. Martin Becker
martin.becker@iese.fraunhofer.de
Contact

Dr. Martin Becker
Fraunhofer IESE
Email: martin.becker@iese.fraunhofer.de
Phone: 0631 / 6800 – 2246

Questions before and after the lecture, via email/phone, and by appointment.
Organisational Issues

lecture (2h) + exercises (1h)

⇒ 4 ECTS credits

**Lecture:** Friday, 15:30-17:00 in IESE

**Exercises:**

- separate time slot, once every two weeks?

**Examinations:** Oral | Written
Exercises

Exercises

- When: Friday, 17:15 - 18:45
- Where: Fraunhofer IESE (Z04.06 – J. Nehmer)

Contact

- Adeline Silva
  Fraunhofer IESE
  E-mail: adeline.silva@iese.fraunhofer.de
## Lectures - Schedule

<table>
<thead>
<tr>
<th>No</th>
<th>Date</th>
<th>Time</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>21-Oct-11</td>
<td>15:30 - 17:00</td>
<td>48-462</td>
</tr>
<tr>
<td>2</td>
<td>28-Oct-11</td>
<td>15:30 - 17:00</td>
<td>IESE</td>
</tr>
<tr>
<td>3</td>
<td>4-Nov-11</td>
<td>15:30 - 17:00</td>
<td>IESE</td>
</tr>
<tr>
<td>4</td>
<td>11-Nov-11</td>
<td>15:30 - 17:00</td>
<td>IESE</td>
</tr>
<tr>
<td>5</td>
<td>18-Nov-11</td>
<td>15:30 - 17:00</td>
<td>IESE</td>
</tr>
<tr>
<td>6</td>
<td>25-Nov-11</td>
<td>15:30 - 17:00</td>
<td>IESE</td>
</tr>
<tr>
<td>7</td>
<td>2-Dec-11</td>
<td>15:30 - 17:00</td>
<td>IESE</td>
</tr>
<tr>
<td>8</td>
<td>9-Dec-11</td>
<td>15:30 - 17:00</td>
<td>IESE</td>
</tr>
<tr>
<td>9</td>
<td>16-Dec-11</td>
<td>15:30 - 17:00</td>
<td>IESE</td>
</tr>
<tr>
<td>10</td>
<td>6-Jan-12</td>
<td>15:30 - 17:00</td>
<td>IESE</td>
</tr>
<tr>
<td>11</td>
<td>13-Jan-12</td>
<td>15:30 - 17:00</td>
<td>IESE</td>
</tr>
<tr>
<td>12</td>
<td>20-Jan-12</td>
<td>15:30 - 17:00</td>
<td>IESE</td>
</tr>
<tr>
<td>13</td>
<td>27-Jan-12</td>
<td>15:30 - 17:00</td>
<td>IESE</td>
</tr>
<tr>
<td>14</td>
<td>3-Feb-12</td>
<td>15:30 - 17:00</td>
<td>IESE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>No</th>
<th>Date</th>
<th>Time</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4-Nov-11</td>
<td>17:15 - 18:45</td>
<td>IESE</td>
</tr>
<tr>
<td>2</td>
<td>18-Nov-11</td>
<td>17:15 - 18:45</td>
<td>IESE</td>
</tr>
<tr>
<td>3</td>
<td>2-Dec-11</td>
<td>17:15 - 18:45</td>
<td>IESE</td>
</tr>
<tr>
<td>4</td>
<td>16-Dec-11</td>
<td>17:15 - 18:45</td>
<td>IESE</td>
</tr>
<tr>
<td>5</td>
<td>6-Jan-12</td>
<td>17:15 - 18:45</td>
<td>IESE</td>
</tr>
<tr>
<td>6</td>
<td>20-Jan-12</td>
<td>17:15 - 18:45</td>
<td>IESE</td>
</tr>
<tr>
<td>7</td>
<td>3-Feb-12</td>
<td>17:15 - 18:46</td>
<td>IESE</td>
</tr>
<tr>
<td>8</td>
<td>10-Feb-12</td>
<td>15:30 - 17:00</td>
<td>IESE</td>
</tr>
</tbody>
</table>
Class Infrastructure

Register via email to adeline.silva@iese.fraunhofer.de ➔ access to Google group
Subject: Register – Lecture
Content
- Name: <your name>
- Course of studies and Semester
- Email
- Experience in Software Engineering
  - University (lectures, classes)
  - Industry
  - Other

Get slides via AGSE Web-Site
Contents of the Lecture

Engineering of variant-rich software / system families
Our Goals

After this course ...

... you will have learned ...

a... what challenges development organizations are facing due to variants
... why opportunistic reuse does not work
... how to systematically reuse software
... methods, techniques, and tools for systematic variation management
Your Expectations?
1. **Software Product Line Engineering: Foundations, Principles and Techniques** by Klaus Pohl, Günter Böckle and Frank J. Linden

2. **Software Product Lines: Practices and Patterns** by Paul Clements and Linda Northrop

3. **Component-Based Product Line Engineering with UML** by Colin Atkinson, Joachim Bayer, Christian Bunse and Erik Kamsties

... and some more research papers
--- Motivation ---
The Beginning

- Most organizations usually do not develop a single system (product), but a set of products in a certain business area
- Many similar products arise over time

- Developing similar products always from scratch is unproductive
  - it costs effort, time and money
  - it leads to redundant effort in maintenance and quality assurance
In the Meantime

- Propagate changes within products
- Propagate changes across products
- Bug fix
Developers versus Maintainers (No Reuse!)
Some more Challenges

- Increasing complexity of systems
- Need for reducing cost, effort, and time-to-market
- Increasing request for quality solutions
- Increasing demand for customized products
- Increasing inter projects/systems dependencies
Problem is well known in the SE-Community:

- Software Crisis (1968)

- You can expect solutions

Problem is not limited to the SE-Community
Approach I: Mature single system engineering

- Improvements of 5-10% per year possible
- Of course you must apply the state-of-the-practice
- You must adopt new approaches from time to time
Approach II: **Reuse**

- Inherent to human nature → natural approach
  1. Use of existing solution
  2. Adapt similar solutions
  3. Develop anew

- Reusing existing solutions
  - saves **time and effort**
  - brings **quality**
  - avoids **complexity** due to replica

Can be applied to any kind of system
Size of Code Base to be Maintained

# delivered systems (to be maintained)

# LoC (to be maintained)

Software Reuse
Reuse approaches: Ad-hoc
Problems with Ad-hoc reuse

Experiences

- **Applied widely:** *Clone and Own*
- **Does not scale** within an organization and across time due to
  - Lacking means for organizing and managing reusable artifacts
    - Search efforts
    - Evaluation efforts
    - Adaptation efforts (80:20 rule holds here)
    - Integration efforts

In most cases a no go!
Reuse Approaches: Domain Engineering

Idea: Proactively develop for reuse
Problems of Domain Engineering

Domain Engineering: Development for reuse
- Understand domain concepts, entities, and relationships
- Set up, maintain, evolve reuse infrastructure

Application Engineering: Development with reuse
- Product development based on large-scale reuse
- Reuse is driven by domain concepts
- No searching for reusable artifacts required

Emphasis is on Domain Engineering
- No clear termination criteria => It takes forever
- Unclear domain boundaries
  - Reusable artifacts become more general or generic than required
  - And thus much harder to reuse and maintain
- Application engineering assumed as requiring no effort (ideal vision)
Domain Engineering – Successes Cases

- GUI Libraries
- Databases
- Middleware
- Operating Systems
- ...

Horizontal, well-understood domains with limited variability

High risk that effort spent in variability support does not pay off
Optimizing Reuse – Product Line Engineering

- Considering the different products an organization or organizational sector delivers as *Product Family* or *Product Line*
- Taking advantage of commonality
- Clear understanding about variability
- Strategic planning of software reuse
- Efficient production

Proactively plan the reuse:
Just the right variability support
More about PLE

Product Line
- Concepts
- Success Stories

... in the next lecture ...

... see you there again