Professor of Computer Science
- For software engineering and open source software
- At the computer science department of the engineering faculty

Previously held research positions at ...
- SAP Labs (Silicon Valley) leading the open source research group
- UBS (Swiss Bank, Zurich) leading the software engineering group

Previously worked in development at ...
- Skyva Inc. (supply chain software, Boston) as software architect
- Bayave GmbH (on-demand business software, Berlin) as CTO
## Engagement Models

<table>
<thead>
<tr>
<th>Recruiting</th>
<th>Student Projects</th>
<th>Shared Projects</th>
<th>Exclusive Projects</th>
<th>Consulting (Limited)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access to students, student jobs</td>
<td>N/A</td>
<td>Researcher</td>
<td>N/A</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Outsourcing</th>
<th>Non-exclusive rights to results</th>
<th>Benefitting from current project, non-exclusive license</th>
<th>Sponsorship of project, possibly exclusive IP rights</th>
<th>Consulting and projects, exclusive IP rights</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-exclusive rights to results</td>
<td>Benefitting from current project, non-exclusive license</td>
<td>Sponsorship of project, possibly exclusive IP rights</td>
<td>Consulting and projects, exclusive IP rights</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Innovation</th>
<th>Non-exclusive rights to results</th>
<th>Benefitting from current project, non-exclusive license</th>
<th>Sponsorship of project, possibly exclusive IP rights</th>
<th>Consulting and projects, exclusive IP rights</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-exclusive rights to results</td>
<td>Benefitting from current project, non-exclusive license</td>
<td>Sponsorship of project, possibly exclusive IP rights</td>
<td>Consulting and projects, exclusive IP rights</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Startups + ecosystem</th>
<th>Startup-Informatik: Improved Ecosystem</th>
<th>Benefitting from current project, non-exclusive license</th>
<th>Sponsorship of project, possibly exclusive IP rights</th>
<th>Consulting for an improved ecosystem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Startup-Informatik: Improved Ecosystem</td>
<td>Benefitting from current project, non-exclusive license</td>
<td>Sponsorship of project, possibly exclusive IP rights</td>
<td>Consulting for an improved ecosystem</td>
<td></td>
</tr>
</tbody>
</table>
Open source enables a development method for software that harnesses the power of distributed peer review and transparency of process. The promise of open source is better quality, higher reliability, more flexibility, lower cost, and an end to predatory vendor lock-in.”

[O18]  See https://opensource.org/about
Definition of Inner Source

“
Inner source refers to the application of the open source approach and benefits to developers within the corporate environment [...]”

Open all artifacts!
Welcome visitors!
Reported Benefits of Inner Source [CR17]

• Higher quality (code) components
  • More users, more eyes, more perspectives: Less bugs faster
    → Ultimately, higher product quality

• Improved knowledge sharing
  • Collaboration across org. boundaries: More weak network ties
    → More innovation

• Higher employee satisfaction
  • More self-determination, broader recognition
    → More attractive employer, better employee retention

• Faster, better, cheaper software development

Example Practitioners of Inner Source
Collaboration in Open Source [R+09]

- **Traditional Work**
  - Hierarchical
    - Closed and hidden silos
    - Assigned to project
  - Status-oriented
    - Public and private discussions
    - Hierarchical status decides
  - Assigned tasks
    - Prescribed process
    - Prescribed jobs

- **Open Collaboration**
  - Egalitarian
    - Open for contribution
    - Everyone can contribute
  - Meritocratic
    - Public discussion process
    - Decisions based on merit
  - Self-organizing
    - People find their own process
    - People find their best project

Communication in Open Source [R15]

1. Public
2. Written
3. Complete
4. Archived

Quality Assurance in Open Source

- Formally: Has commit (write) rights
- Performs bulk of work; quality assurance
- Provides small features, bug fixes
- Submits patches (no commit rights)
- Knows and uses software
- Helps with comments, feedback
Example Scenarios of Inner Source Collaboration

- Bug fix
- Refactoring
- New component
Inner Source at SAP

- Inner source (then “firm-internal open source”)
  - Established to improve openness mindset, tear down silos
  - Established to simplify development using a software forge

- Set-up a software forge called SAP Forge
  - Set-up forge in 2006 (joint work of SAP Labs Silicon Valley and Israel)
  - Managed growth from 2006-2009 from 0 to 1500 registered developers

- Evolution of SAP Forge
  - Continued linear growth even after I left
  - Documented experiences in [R+09]
Welcome to SAP Forge!

SAP Forge is a collaboration system that was set up in order to encourage cooperative work within SAP. Why? Because the success of most projects nowadays depends on the quality of collaboration between people. This includes internal collaboration among project members and external collaboration with other groups.

SAP Forge provides tools for increased transparency and improved collaboration. It makes it easier for people to collaborate and accumulate shared knowledge, even when they are spread across multiple sites, countries, and time zones.

Specifically, SAP Forge provides:
- A single point of entry for all project-related information
- Message boards / discussion forums
- Issue trackers for bugs, feature requests, patches, etc.
- Task management
- Mailing lists
- Website hosting
- Permanent archival for file releases
- Document management
- News & announcements
- Surveys
- Source Code Management (SCM) repositories like CVS and Subversion (Perforce integration is being planned)
- Full backup
- Administration

Participation

SAP Forge will succeed with participation. This will allow people to find all the useful information they need. Please participate by:
- Registering to use the system
- Contributing to discussions
- Reporting issues
- Providing feedback
Properties of a Software Forge

- Projects are
  - Available in one place
  - Findable / searchable
  - Better documented
  - Available forever
The Mobile Retail Demo Project

• Benefits to project
  • Found 18 volunteers (18/27)
  • These volunteers
    − Were motivated
    − Had broad expertise
    − Improved requirements
    − Brought broad support
    − Helped with testing
    − Increased project visibility
    − Aided display of significance

• Finally, transfer of project to product division

• Benefits to company
  • Project uses common tools
  • Tool set-up is understood
  • Project can be found
  • Project is archived
Final Reflections on SAP Forge / Inner Source at SAP

- Steady progress and interest
  - From 0 to 1500 developers in three years

- Developers often just wanted the tools
  - At the click of a button rather than lots of forms

- Approach worked well for research-to-product transfer
  - SAP Forge gave research early exposure to products

- A tool alone cannot create cultural change
  - But can be very helpful to it
Key Facts about Cases

- **Product lines**
  - Organization
    - Platform = cost center
    - Products = profit centers
  - Domain
    - Business software
    - Healthcare software
    - Telco carrier software
  - Developers
    - More than 500 developers each
    - Culturally and socially homogeneous
    - All in one location, i.e. no GSD problems

- **Companies**
  - Large international firms
  - All older than 20 years
  - Diversified, many product lines
## Case Access and Materials Gathered

<table>
<thead>
<tr>
<th></th>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Unit of analysis access</strong></td>
<td>Direct access to all units of analysis</td>
<td>Direct access to all units of analysis</td>
<td>Mediated by sponsor</td>
</tr>
<tr>
<td><strong>Subject access</strong></td>
<td>Interview partners selected by consensus</td>
<td>Interview partners selected by consensus</td>
<td>Mediated by sponsor</td>
</tr>
<tr>
<td><strong>Types of data collected</strong></td>
<td>Collateral materials, interview notes</td>
<td>Collateral materials, interview notes</td>
<td>Collateral materials, interview notes</td>
</tr>
<tr>
<td><strong>Researchers</strong></td>
<td>Two researchers (one interviewer, one scribe)</td>
<td>Two researchers (one interviewer, one scribe)</td>
<td>Single researcher taking his own notes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case</th>
<th>Year</th>
<th>No. Interviews</th>
<th>Workshops</th>
<th>Supplemental Materials</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2012</td>
<td>11</td>
<td>5</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>2013</td>
<td>6</td>
<td>None</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>2013</td>
<td>4</td>
<td>3</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Problems in PLE 1 / 2

Problems from and with Organizational Structure

- Lack of global business unit perspective
  - 1, 2, 3
- Insufficient trust between product units
  - 1, 2, 3
- Limited understanding of needs of other organizational units
  - 1, 2, 3

Separation of product units as profit centers
- 1, 2, 3

Power play between product units
- 1, 2, 3

Insufficient intra-organizational-unit collaboration
- 1, 2, 3

Insufficient standardization on tools, methods, and processes
- 1, 2, 3

Insufficient developer networking
- 1, 2

Lack of resources at platform organization
- 1, 2, 3

Lack of time to act beyond immediate needs
- 1, 2
Problems in PLE 2 / 2

Problems from and with Organizational Structure

- Lack of resources at platform organization
  
  1, 2, 3

- Power play between product units
  
  1, 2, 3

- Limited understanding of needs of other organizational units
  
  1, 2, 3

- Insufficient intra-organizational-unit collaboration
  
  1, 2, 3

- Insufficient standardization on tools, methods, and processes
  
  1, 2, 3

Problems with Domain Engineering Process

- Poorly prioritized domain requirements
  
  1, 2, 3

- Unclear (to platform) reusable asset requirements posed by product units
  
  1, 2, 3

- Insufficient reusable asset quality
  
  1, 2, 3

- Insufficient reusable asset extraction process
  
  1, 2, 3

- Too much copy and paste
  
  1, 2, 3

- Insufficient product line architecture
  
  1, 2

Undesired Effects

- Delayed domain artifact realization
  
  1, 2, 3

- Rework and wasted effort
  
  1, 2, 3

- Insufficient reusable asset quality
  
  1, 2, 3

- Insufficient reusable asset extraction process
  
  1, 2, 3

- Too much copy and paste
  
  1, 2, 3

- Insufficient product line architecture
  
  1, 2

- Delayed product delivery
  
  1, 2, 3

- Increased defect rate
  
  1, 2, 3

- Too many redundant software components
  
  1, 2, 3

General Problems

- Insufficient underlying technology
  
  1, 2
The separation of product units as profit centers from a platform organization as a cost center, leads to delayed deliveries, increased defect rate, and redundant software components.

Problems with Inner Source 1 / 2

- (Some) middle managers
  - Fear transparency and loss of control
  - Fear not meeting performance goals

- “Most managers dislike showing their planning documents widely; it might open them up for critique.” Manager (product unit)

- “Allowing developers to contribute to inner source projects may feel like losing control to some managers.” Developer (platform)

- “Managers may disallow contribution to inner source if they feel their own product is not benefiting enough.” Developer (platform)
Problems with Inner Source 2 / 2

- (Some) software developers
  - Dislike performing quasi-public work
  - Fear follow-on and maintenance work

- “Inner source leads to [public] mistakes, and [some] developers fear mistakes because they lead to reputation loss among colleagues.” Manager (platform)

- “Most developers hate maintenance of important components because it makes them responsible for fixing high-priority bugs; this creates too much stress.” Developer (product unit)
Establishing Successful Inner Source

- Inner source tooling
  - Software forge
  - Peer review
  - ...

- Inner source projects
  - Open collaboration
  - Open communication
  - ...

- Inner source programs
  - Cultural change
  - Managerial incentives
  - ...

Diagram:
- P (roposal)
- I (ncubation)
- D (evelopment)
- M (aintenance)
- OSS
- EOL
Five Years Later, Revisiting One Case ... [C+18]

Platform of reusable assets – but no dedicated platform organization
Management Accounting for Inner Source

core-drivers (*Inner Source Project*)

- **Number of Patches**
- **Number of Contributors**
- **Number of Involved Business Units**

- **Project Health Score (1 - 10)**
  - *Good health!*

- **Top Contributors**

- **Top Business Units**
Transfer Pricing

- How to value intellectual property on a fine-grain scale (commits)?
- How to optimize (legally) the resulting IP flow for tax purposes?
Typical Inner Source Work Packages 1 / 2

- Pilot project
  - Situation analysis
  - Process definition, tool selection
  - Pilot project selection
  - Project goal definition
  - Pilot project support
  - Lessons learned

- Support measures
  - Educational materials
  - Workshops
  - Literature
Typical Inner Source Work Packages 2 / 2

- Program support
  - Situation analysis

- Program definition
  - Charter definition
  - Process definition
  - Tool selection

- Program implementation
  - Educational measures
  - Workshops
  - Information exchange
  - Benchmarking

- Project consulting
  - Situation analysis
  - Project support
  - Lessons learned
Thank you! Questions?

dirk.riehle@fau.de – http://osr.cs.fau.de

dirk@riehle.org – http://dirkriehle.com – @dirkriehle