Praktische Softwaretechnik

Prof. Dr. Dirk Riehle

Lecture 23/30 of 20.01.2010

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Table of Contents

TECHNISCHE FAKULTÄT

Friedrich-Alexander-Universität Erlangen-Nürnberg

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

The Three Main Process Frameworks

Plan-Driven

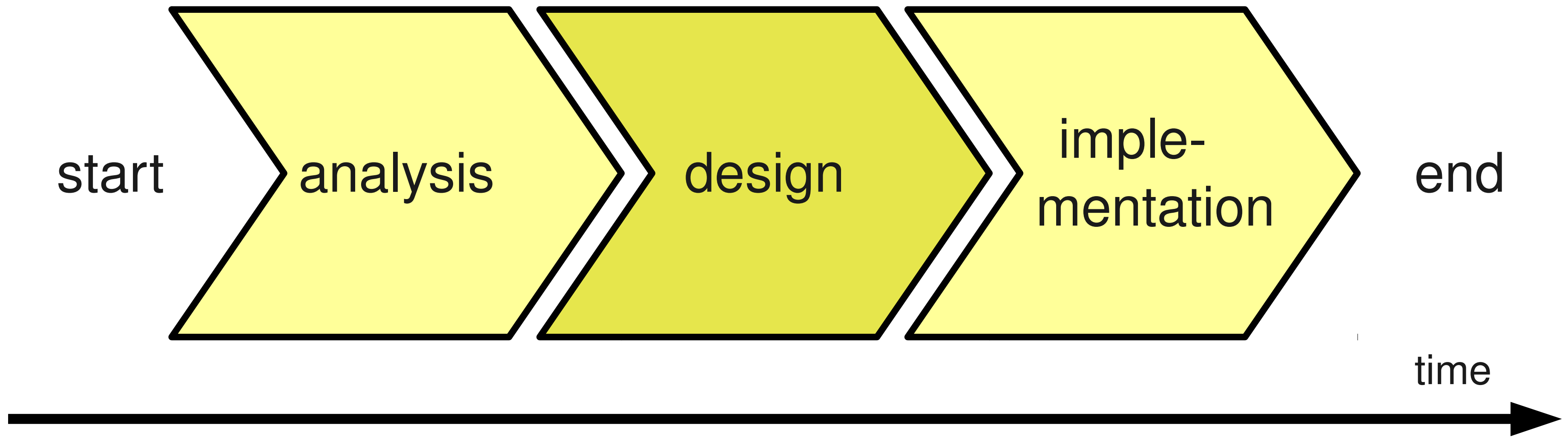
Agile Methods

Open Source

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Plan-Driven Software Development

- Linear, phase-oriented process
  - Main goal is to minimize risk through careful upfront planning
  - Equates phases with activities
  - No iterations, just one pass



Friedrich-Alexander-Universität Erlangen-Nürnberg

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

An Illustration of Plan-Driven Development

Picture credit: From http://www.megawoosh.com

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Example Phases in Plan-Driven Development

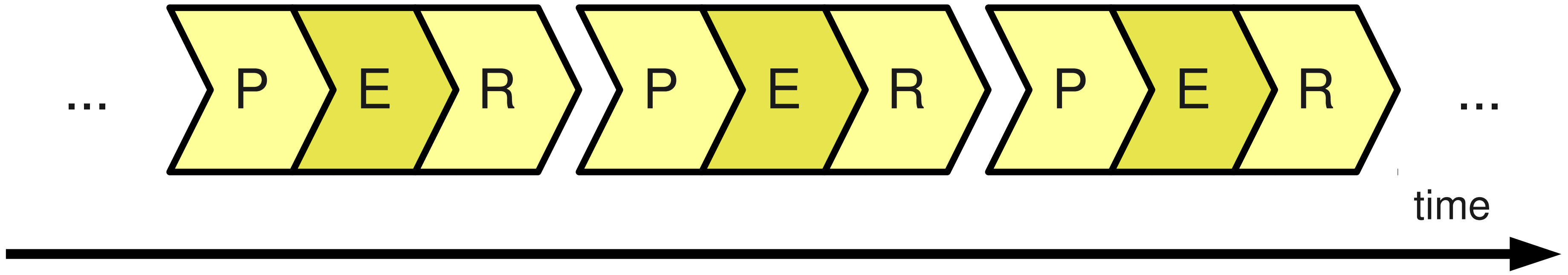
- Project negotiations
- Requirements analysis
- Contract definition
- System analysis
- Software architecture design
- System design
- User interface design
- Implementation
- System test
- Acceptance test
- Handover
- Actual use
- Lawsuit

no feedback until after delivery

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Agile Methods

- Agile methods is the name of a class of process frameworks
  - Extreme Programming, SCRUM, DSDM, Adaptive Software Development, Crystal, Feature-Driven Development, Pragmatic Programming, etc.
  - Unified by the recognition of a common philosophical base and joined in their rejection of the traditional life-cycle model



Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Agile Philosophy and Agile Manifesto

- Core agile method values
  - **Individuals and interactions** over processes and tools
  - **Working software** over comprehensive documentation
  - **Customer collaboration** over contract negotiation
  - **Responding to change** over following a plan

- Codified (and marketed) as **the Agile Manifesto**
  - See http://www.agilemanifesto.org

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Two Example Agile Methods

Scrum

(mostly as process framework)

Extreme Programming (XP)

(mostly for development practices)

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

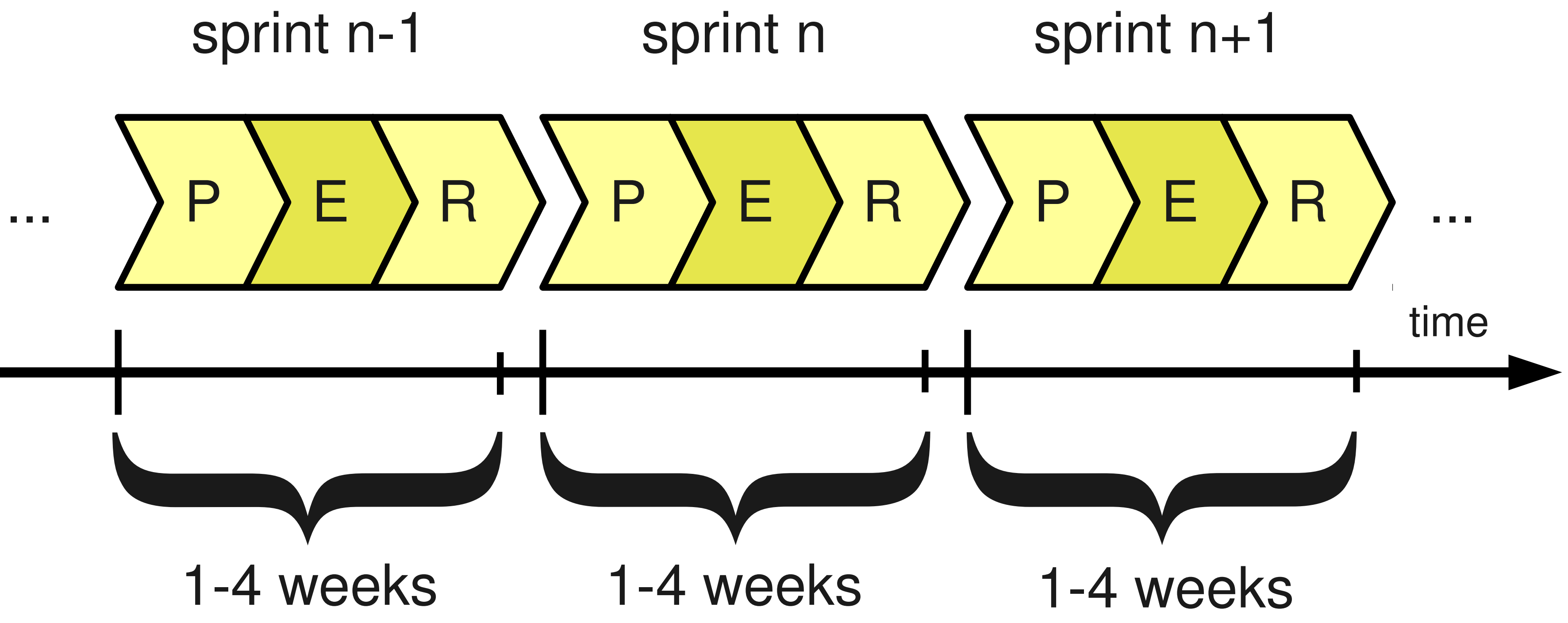
Process and Programming Practices

User Stories Feature Prioritization Release Planning	Sprint Planning Sprint Review ...
Test First Pair Programming Continuous Integration	Daily Release Refactoring ...

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Scrum's Development Process

- Succession of **equal-length** sprints (= short iterations)
- Intervention points are during planning and review
- Product owner is always available to answer questions



P = planning - E = engineering - R = review (or release or retrospective)

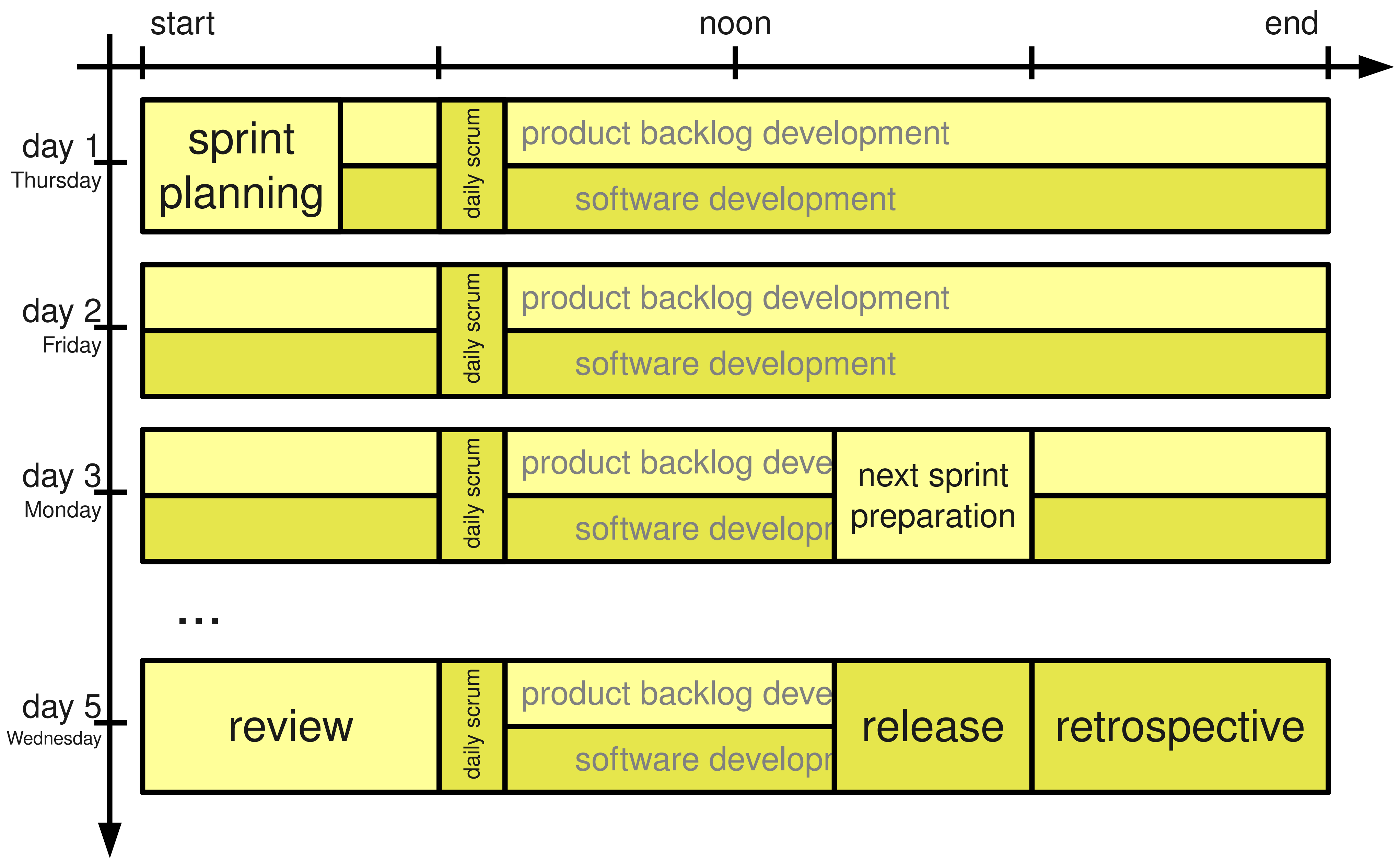
Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

The Significance of (Short) Iterations

Establishes shortest economically feasible feedback loop

- Short iterations lead to focus on high-value features first
- User feedback helps team steer product to meeting needs right
- Quality feedback helps team deliver high-quality software
- Feedback loop ensures problems and hence risk surfaces early
- Iterations help recognize and realize new innovative features
- Established well-worn rhythm is sustainable, avoids burnout
- Even with missed deadlines, partial functionality is better than none

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg


start
noon
end
day 1
Thursday
day 2
Friday
day 3
Monday
...
day 5
Wednesday
sprint planning
daily scrum
product backlog development
software development
daily scrum
product backlog development
software development
daily scrum
product backlog deve
software developr
next sprint preparation
review
daily scrum
product backlog deve
software developr
release
retrospective

Test-Driven Development

- Test-driven development (abbrev. TDD)
  - Is a core process for programming activities
  - Promises minimal fluctuations in development speed
  - Takes the stress out of refactoring your code base
  - Considered a high-end practice (requires too much discipline for many)
  - Requires significant experience with ancillary tools and techniques
- First described by Kent Beck in [Beck 2003]

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

TDD Follows the XP Mantra

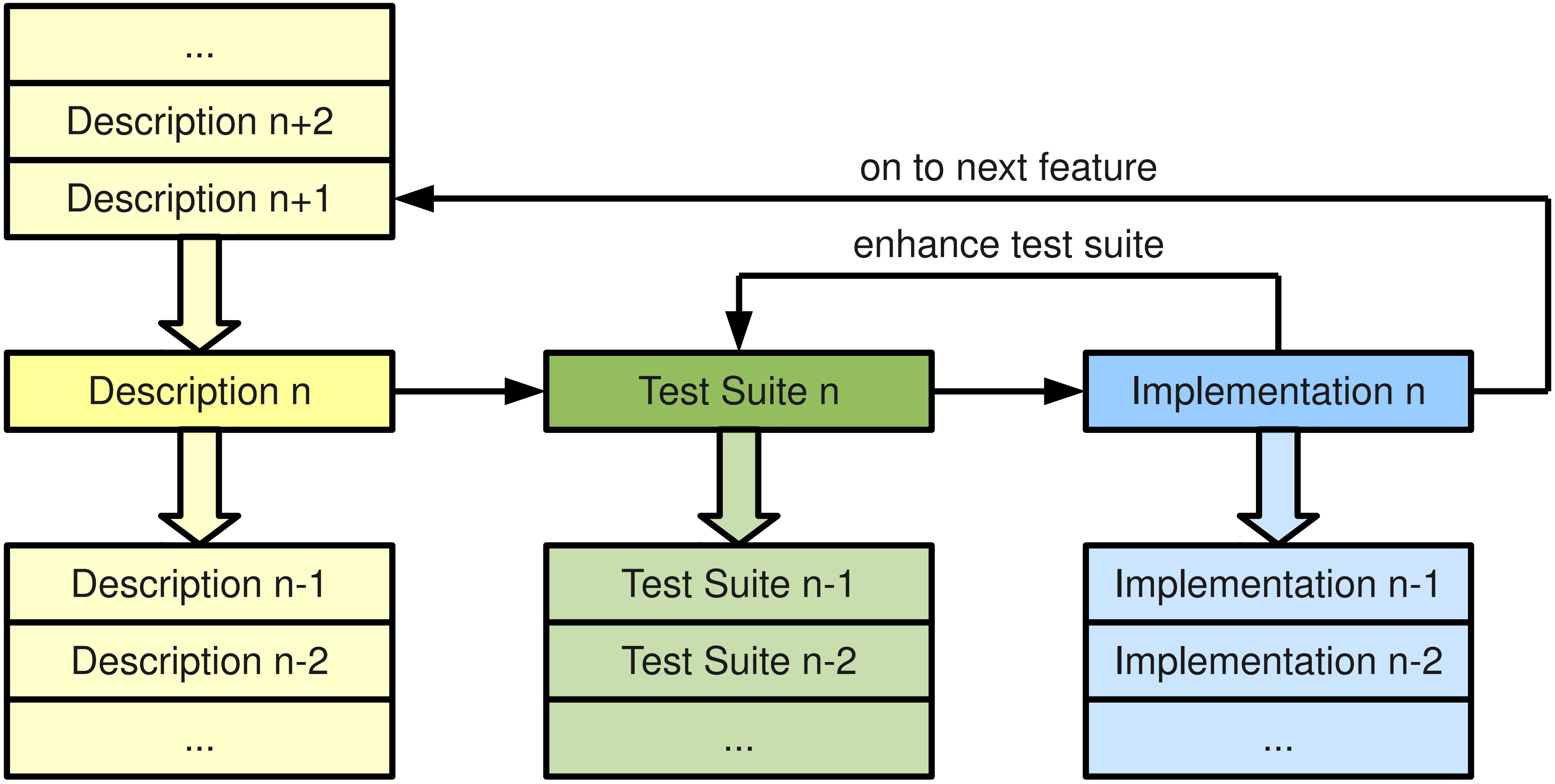
Make it run

Make it right

Make it fast

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

The Smallest Possible (Believable) Process



Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

The Promise of Test Driven Development

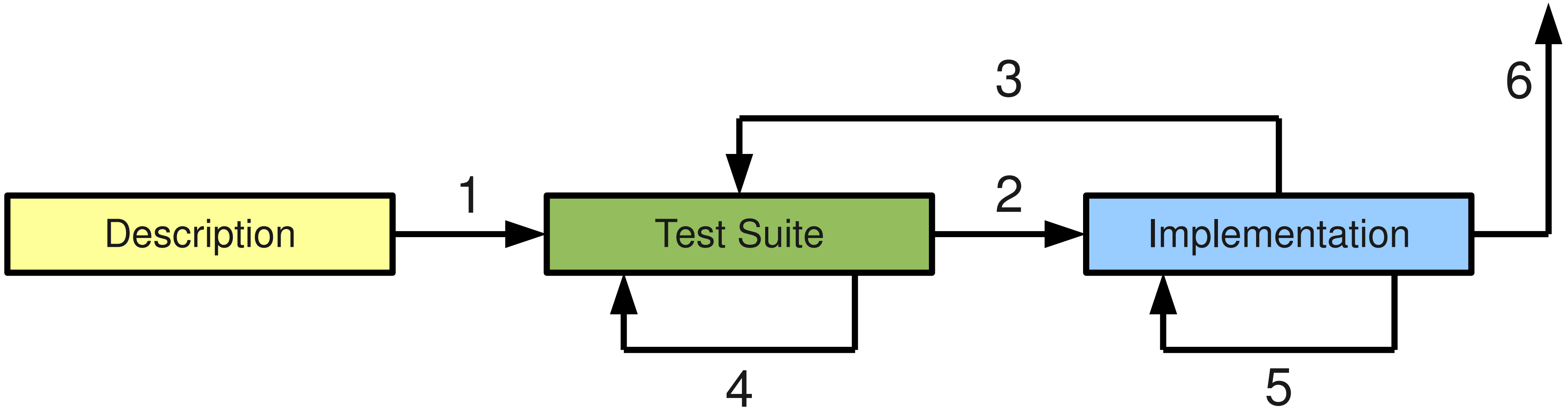
- You can release after every feature implementation
- The likelihood of failure or bugs is as minimal as possible
- The product grows incrementally and steadily
- The code is of pristine (and lean) quality

- **TDD promises minimal fluctuations in development speed**

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

TDD Subprocesses

1. Translate description into test suite
2. Implement feature to fulfill ("green-bar") test suite
3. Revise test suite from new domain and implementation insights
4. Refactor test suite to keep code healthy
5. Refactor implementation to keep design and code healthy
6. Exit when test suite is complete and all tests succeed



Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

TDD is a Complex Adaptive System

Two simple rules

only write new code when a test fails

eliminate waste

deliver

clean code that works

Winter Semester 2009/2010 Lecture · Praktische Softwaretechnik
Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Definition of Open Source (Abbreviated)

- Open source software is software that is
  - available in source code form
  - can be freely used and modified
  - can be freely redistributed
  - cf. four freedoms of software
- The open source initiative
  - maintains the definition and trademark
  - approves open source licenses

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Some Open Source Projects

APACHE

SUGARCRM
COMMERCIAL OPEN SOURCE

OpenOffice.org

MySQL®

JASPERSOFT

eclipse
THE ECLIPSE PROJECT

PostgreSQL

debian

python

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Free/Libre/Open Source Software

- The free software movement
  - Initiated (and still led) by Richard Stallman (MIT) in the 1980 to free software (from being closed)
  - Free software philosophy summarized as “free as in 'free speech' not as in 'free beer'”
- Free Software Foundation
  - http://www.fsf.org
  - U.S. 501(c) non-profit organization

- The open source movement
  - Eric Raymond: “Cathedral and the Bazaar” --- describes open source as development method
  - Formalized 1998 to address perceived anti-commerce bias of “free software”
- Open Source Initiative
  - http://www.opensource.org
  - U.S. 501(c) non-profit organization

FLOSS = Free/Libre/Open Source Software

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

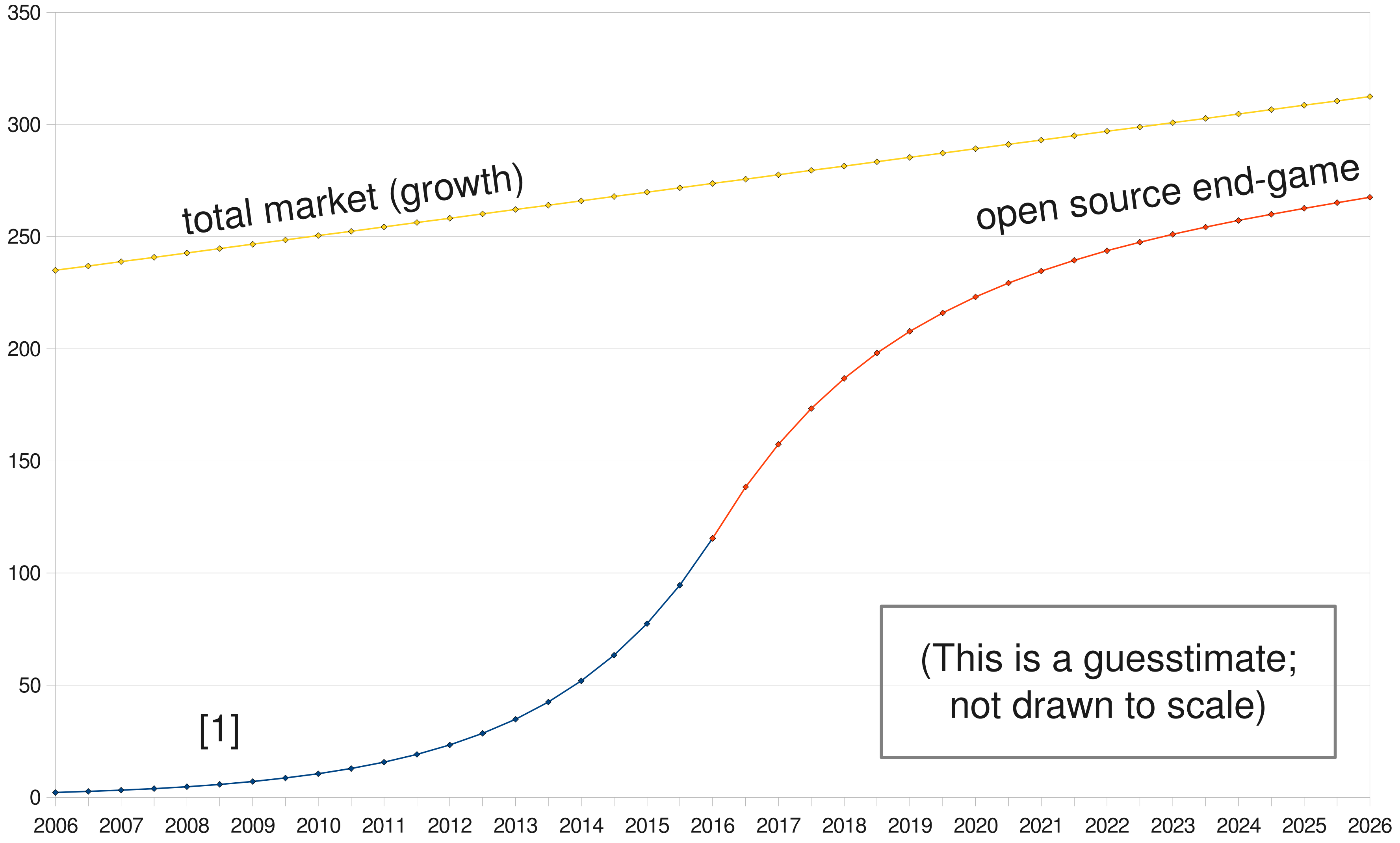
Economic Impact of Open Source

- Commercial use
  - Gartner: “By 2012, more than 90 percent of enterprises will use open source [...]” [1]
  - By and large, open source has gone mainstream, is just a product like any other
  - Today, formal open source adoption strategy elusive and TCO gains unclear
  - Open source dominates software-as-a-service and in startups

- Packaged software market
  - In 2006, open source held \$1.8B / \$235B = 0.8% of market revenue [2] [3]
  - IDC: Open source revenue will reach \$5.8B by 2011 (26% CAGR 2006-11) [2]

[1] Gartner Inc. “The State of Open Source 2008.”
[2] IDC. “Worldwide Open Source Software Business Models 2007–2011 Forecast: A Preliminary View.”
[3] Software & Information Industry Association. “Packaged Software Industry Revenue and Growth, 2006.”

Current and Projected Growth Pattern



(This is a guesstimate; not drawn to scale)

[1] Deshpande, Riehle. "The Total Growth of Open Source." In Proceedings of OSS 2008.

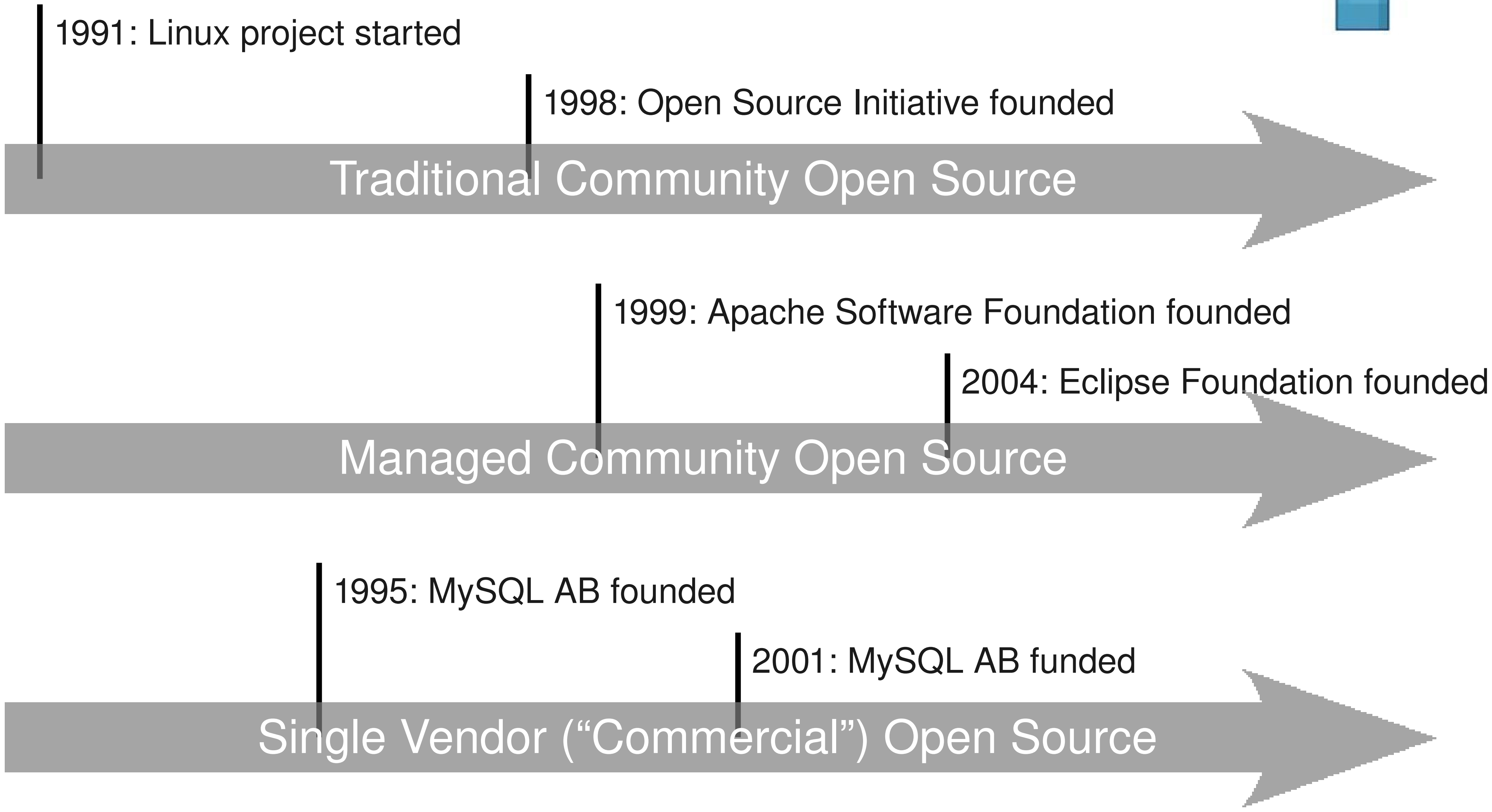
Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Types of Open Source Projects

	Community-owned	Single or dominant proprietor
Single product or product line	**Community Open Source** (e.g. Linux, TikiWiki)	**Commercial Open Source** (e.g. MySQL, Jasper)
Multi-product assembly (“stack”)	**Community Distribution** (e.g. Debian)	**Commercial Distribution** (e.g. RHEL, SLES)

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Time-line of Open Source


1991: Linux project started
1998: Open Source Initiative founded
Traditional Community Open Source
1999: Apache Software Foundation founded
2004: Eclipse Foundation founded
Managed Community Open Source
1995: MySQL AB founded
2001: MySQL AB funded
Single Vendor ("Commercial") Open Source

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Open Collaboration [1]

Egalitarian

Meritocratic

Self-organizing

[1] Dirk Riehle et al. “Open Collaboration.” IEEE Software vol. 26, no. 2 (March/April 2009).

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Traditional vs. Open Collaboration

Traditional Work

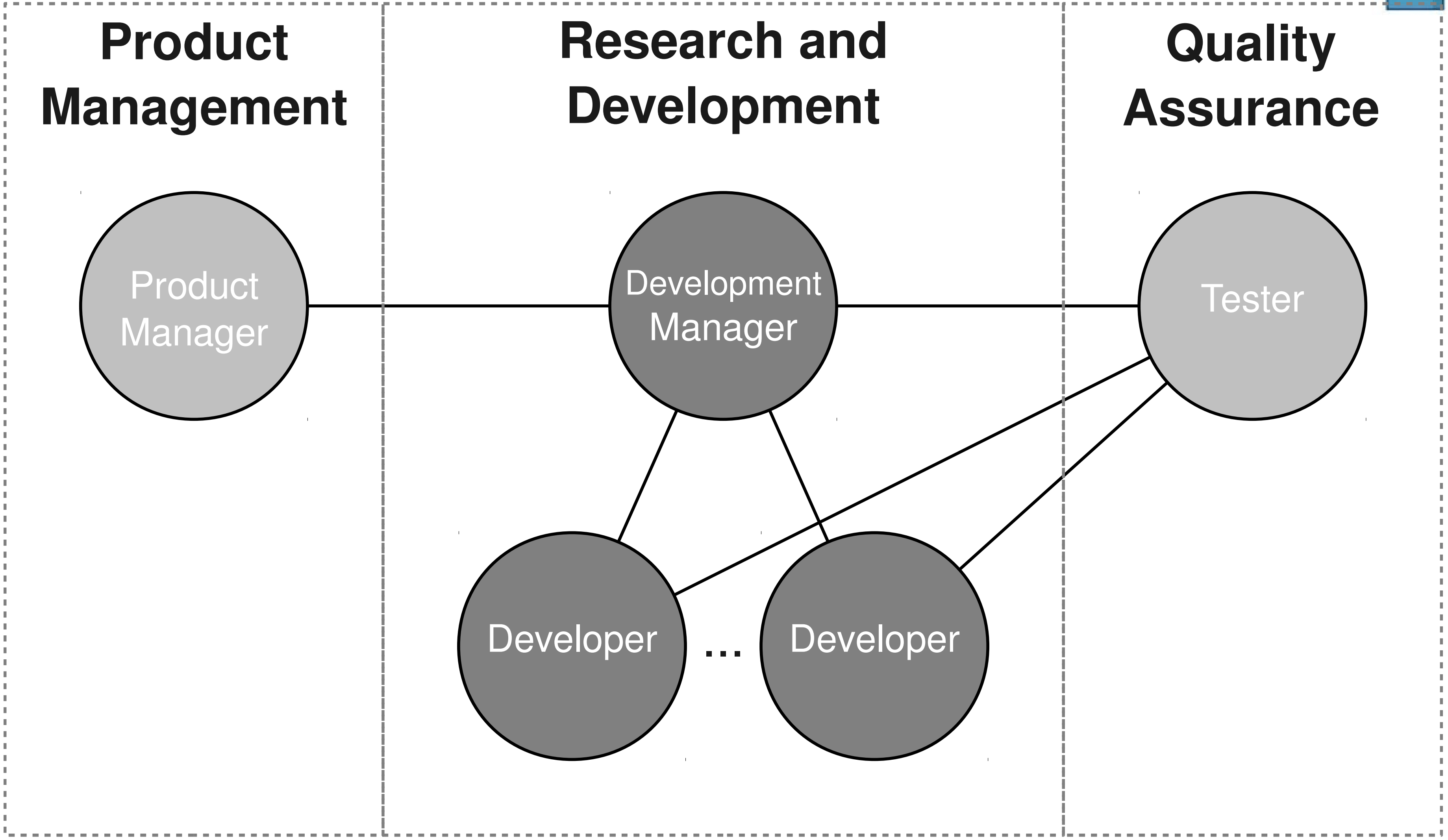
- Hierarchical
  - Closed and hidden silos
  - Assigned to project
- Status-driven
  - Public and private discussions
  - Hierarchical status decides
- Assigned tasks
  - Prescribed process
  - Prescribed jobs

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

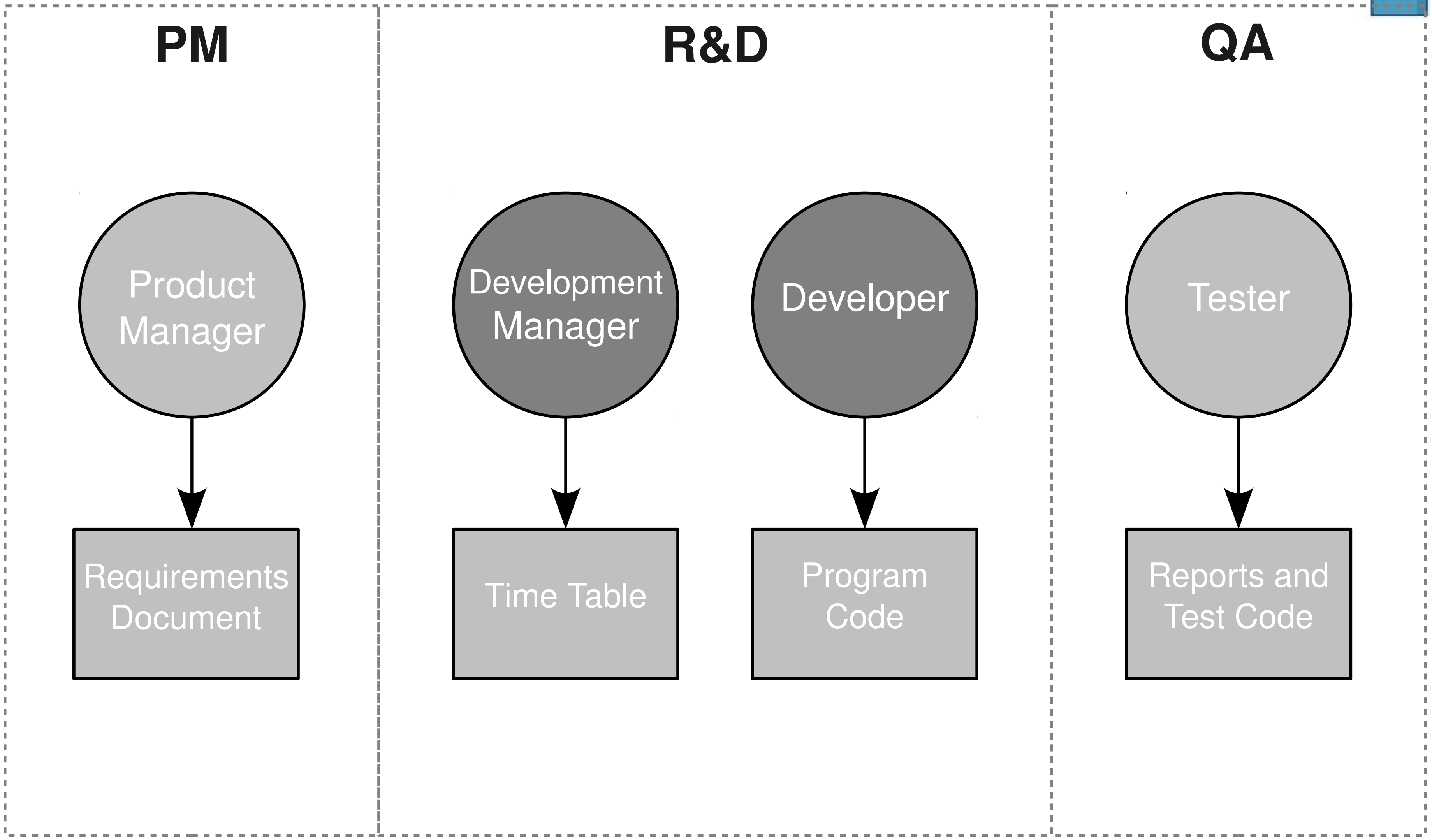
Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Roles in Traditional Processes



Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

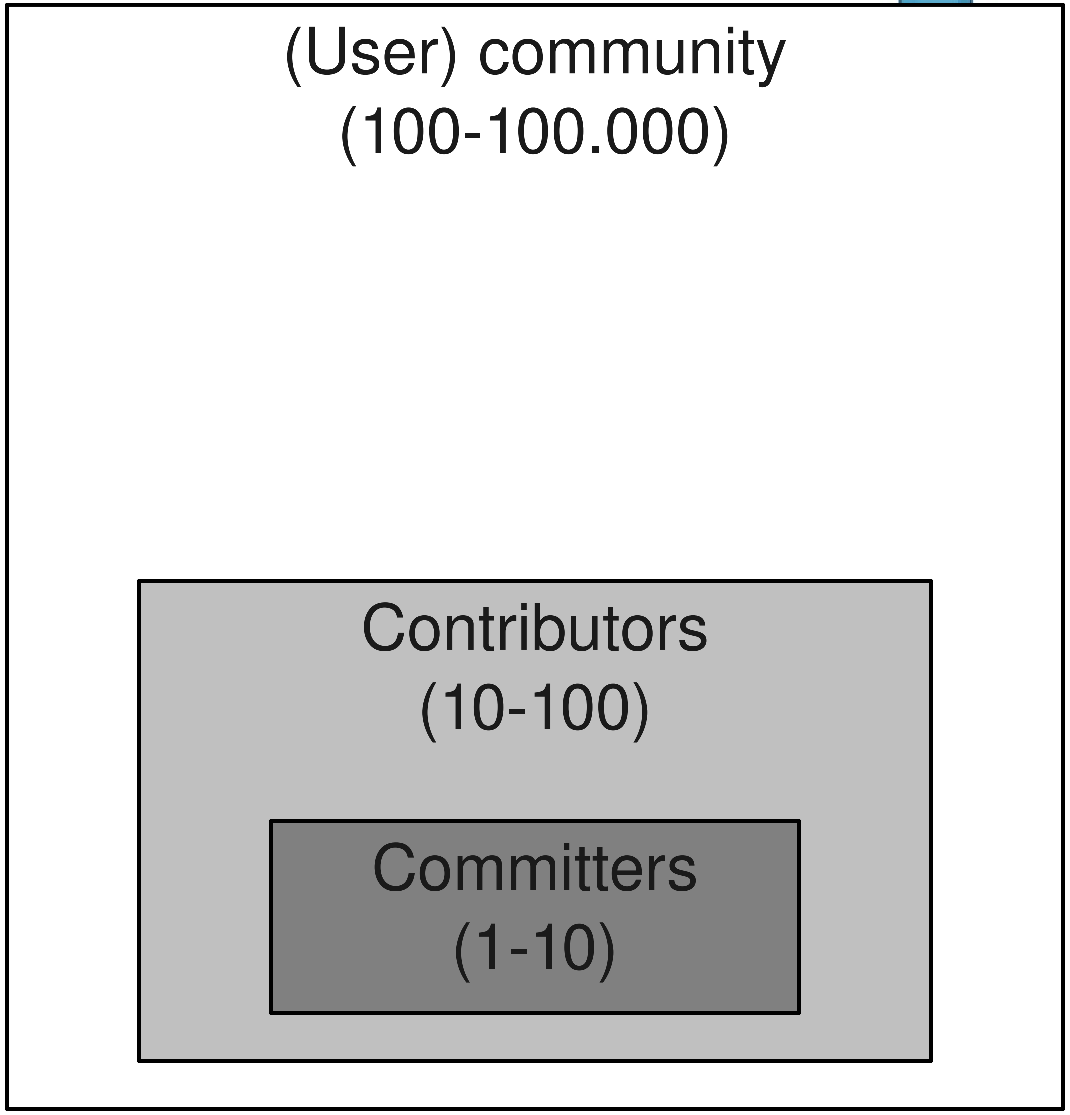
Artifacts in Traditional Processes



Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Committers, Contributors, Community

- Committers
  - Main developers
  - With access rights
- Contributors
  - Casual developers
  - Submit patches
- (User) community
  - Provide bug reports
  - Provide feedback



Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

PostgreSQL Product Management

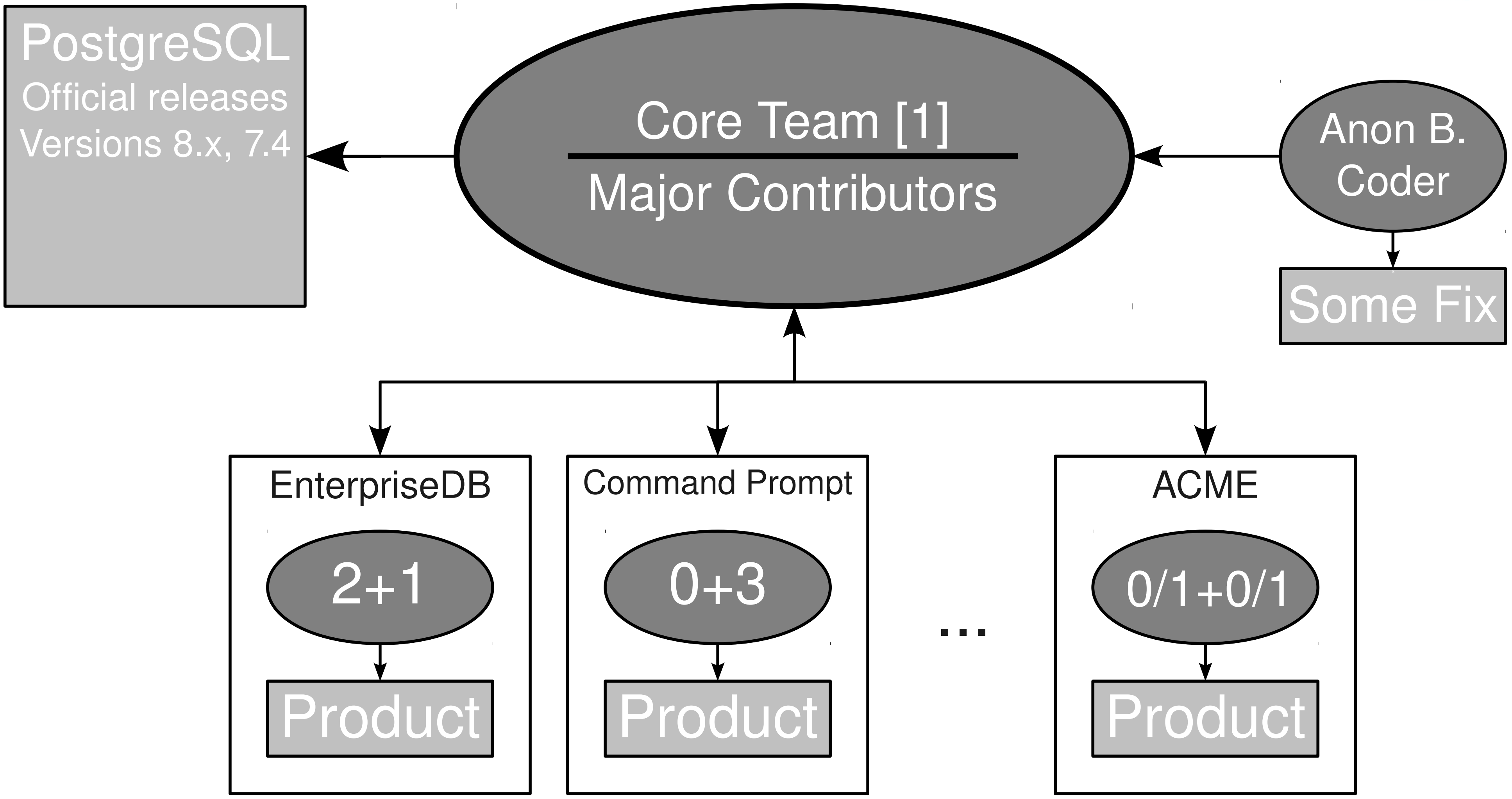
- Who defines features? [1]
  - Patches
  - Scratching one's itch
  - Individual developer initiatives
  - Company-driven contributions
  - User polls on project home page

- Product management practices
  - Leadership by core team
  - Maintenance of a todo list [2]
  - No prioritization: “Pick your feature”

[1] http://www.postgresql.org/developer/roadmap
[2] http://wiki.postgresql.org/wiki/Todo

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

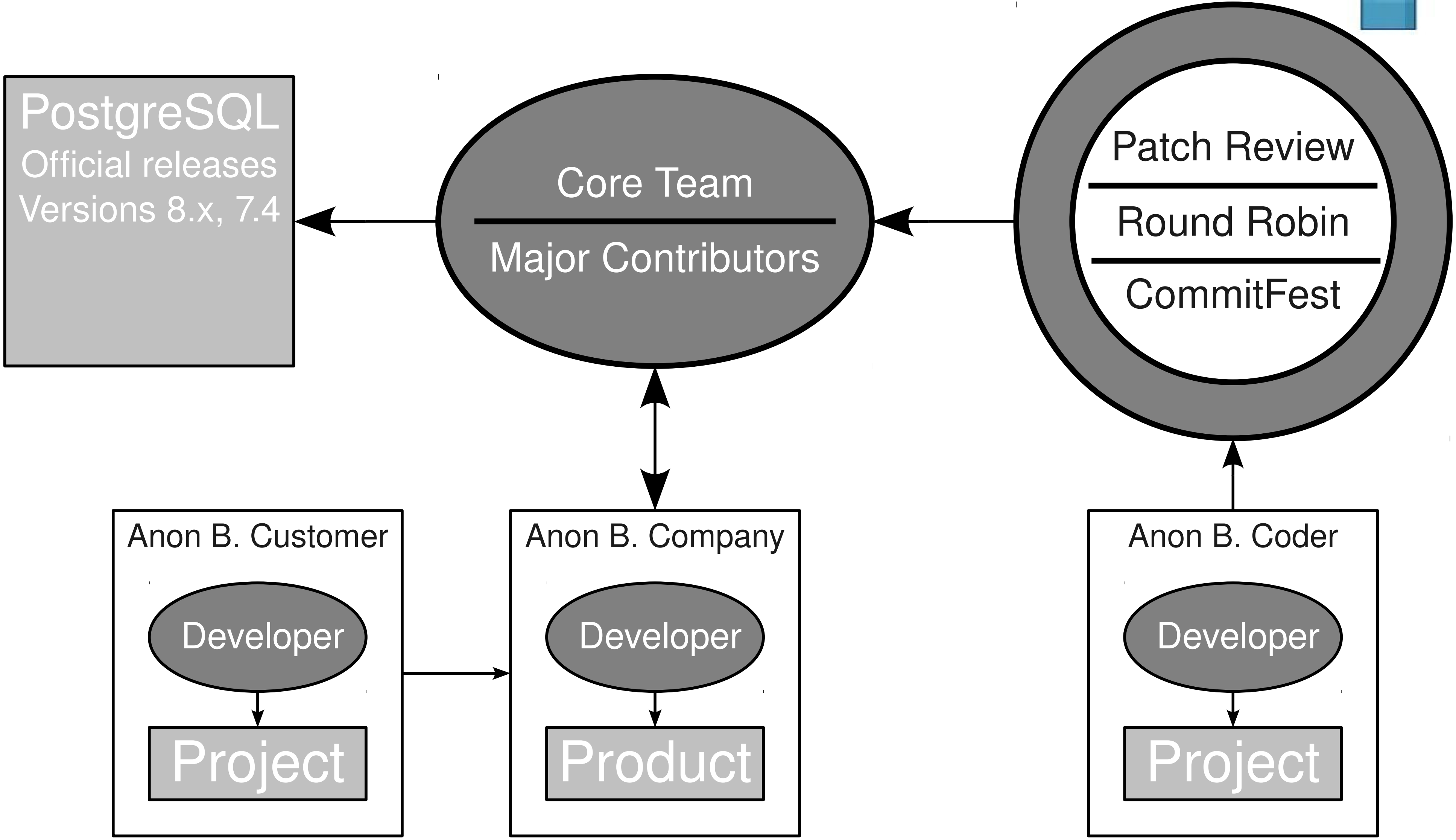
PostgreSQL Development



[1] http://www.postgresql.org/community/contributors/

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

PostgreSQL Quality Assurance



Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Comparison of Process Frameworks

		new application domain	
		no	yes
large project	no	plan-driven agile methods open source	**agile methods** open source
	yes	**plan-driven** open source	agile methods **open source**

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg

Questions? Feedback!

http://dirkriehle.com - dirk.riehle@cs.fau.de - @dirkriehle

Copyright © 2010 Dirk Riehle · Open Source Research Group · Friedrich-Alexander-University of Erlangen-Nürnberg