Increasing Software Quality using the Provenance of Software Development Processes

Andreas Schreiber <andreas.schreiber@dlr.de>

German Aerospace Center (DLR)

Berlin / Braunschweig / Cologne

Outline

- Introduction
- Provenance
- Software Development Processes
- Queries

Introduction

Problem

- Today's software development processes are complex
- Massive interaction between developers and tools as well as between tools (manually or automatically)
- Tracing and understanding the process is hard
- Software isn't reused because of lack of trust and quality

Solution

- Recording of process information during runtime
- Analysis of recorded information for insight and confidence

Standardized (W3C) solution: Provenance

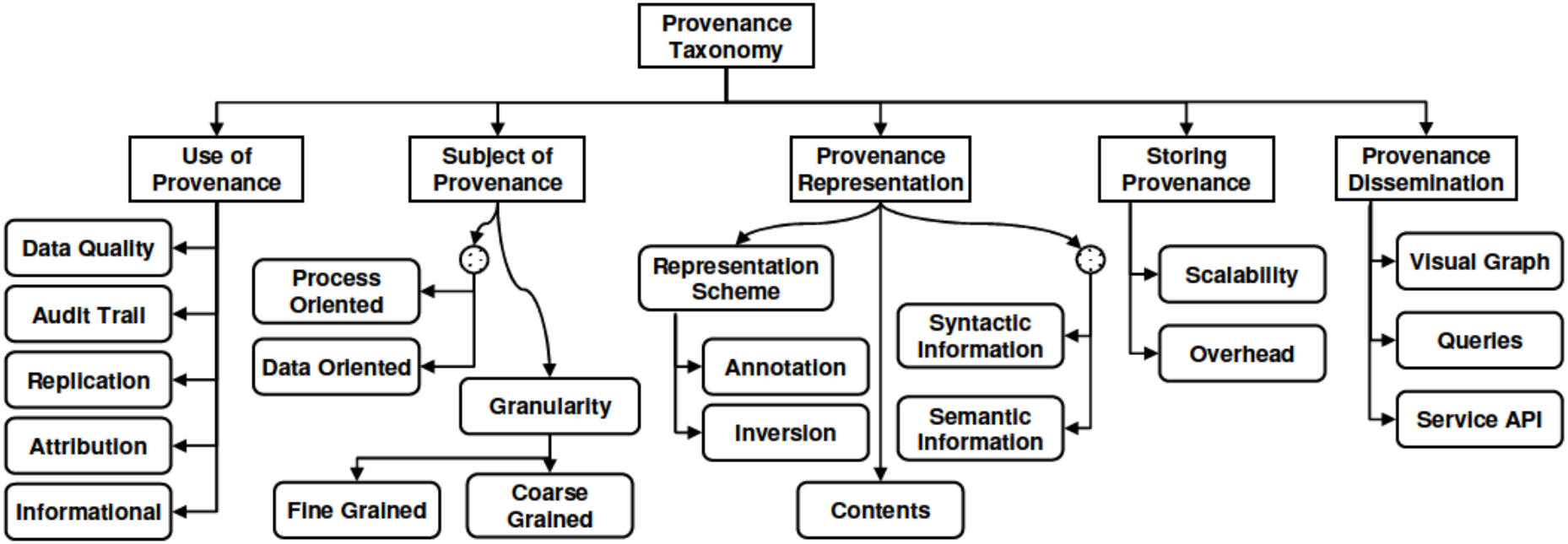
Provenance Definition

Provenance is defined as a record that describes the people, institutions, entities, and activities involved in producing, influencing, or delivering a piece of data or a thing.

(W3C Provenance Working Group, http://www.w3.org/2011/prov)

Provenance
Research Area Since 2002

- Luc Moreau. The foundations for provenance on the web. *Foundations and Trends in Web Science*, November 2009.
- Simmhan, Yogesh L., Beth Plale, and Dennis Gannon: A survey of data provenance in e-science.



Provenance Application Areas

General Areas

- Information systems: Origin of data, who was responsible for its creation
- Science applications: How the results were obtained
- Publications: Origins and references of published results

Applications involve

- Engineering
- Finance
- Security
- Climatology & earth sciences
- Medicine, pharmacy & biomedicine
- Software Development

http://www.w3.org/2011/prov/wiki/ISWCProvTutorial

Provenance Goal

Express special “meta” information on the data

- Who played what role in creating the data
- View of the full revision chain of the data
- In case of integrated data, which part comes from which original data and under what process

http://www.w3.org/2011/prov/wiki/ISWCProvTutorial

Realizing Provenance

Provenance requires a complete model

- Describing the various constituents (actors, revisions, etc.)
- Balance between
  - simple (“scruffy”) provenance: easily usable and editable
  - complex (“complete”) provenance: allows for a detailed reporting of origins, versions, etc.

http://www.w3.org/2011/prov/wiki/ISWCProvTutorial

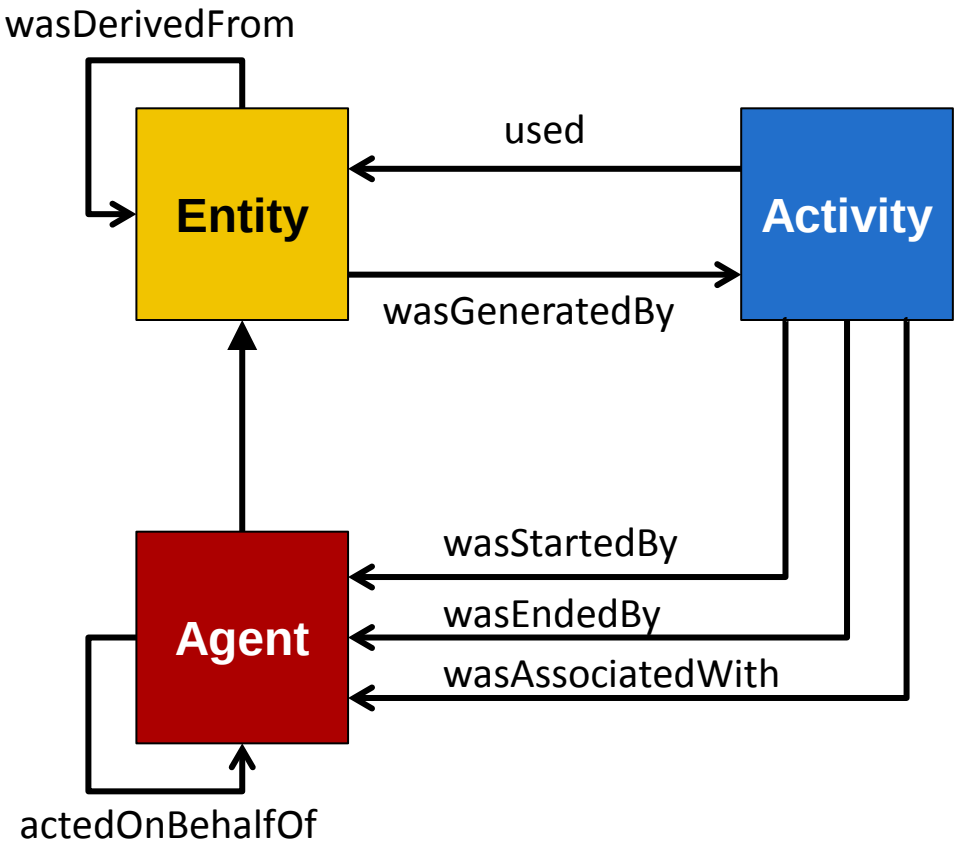
W3C Provenance Data Model (PROV-DM)
Concepts

Nodes

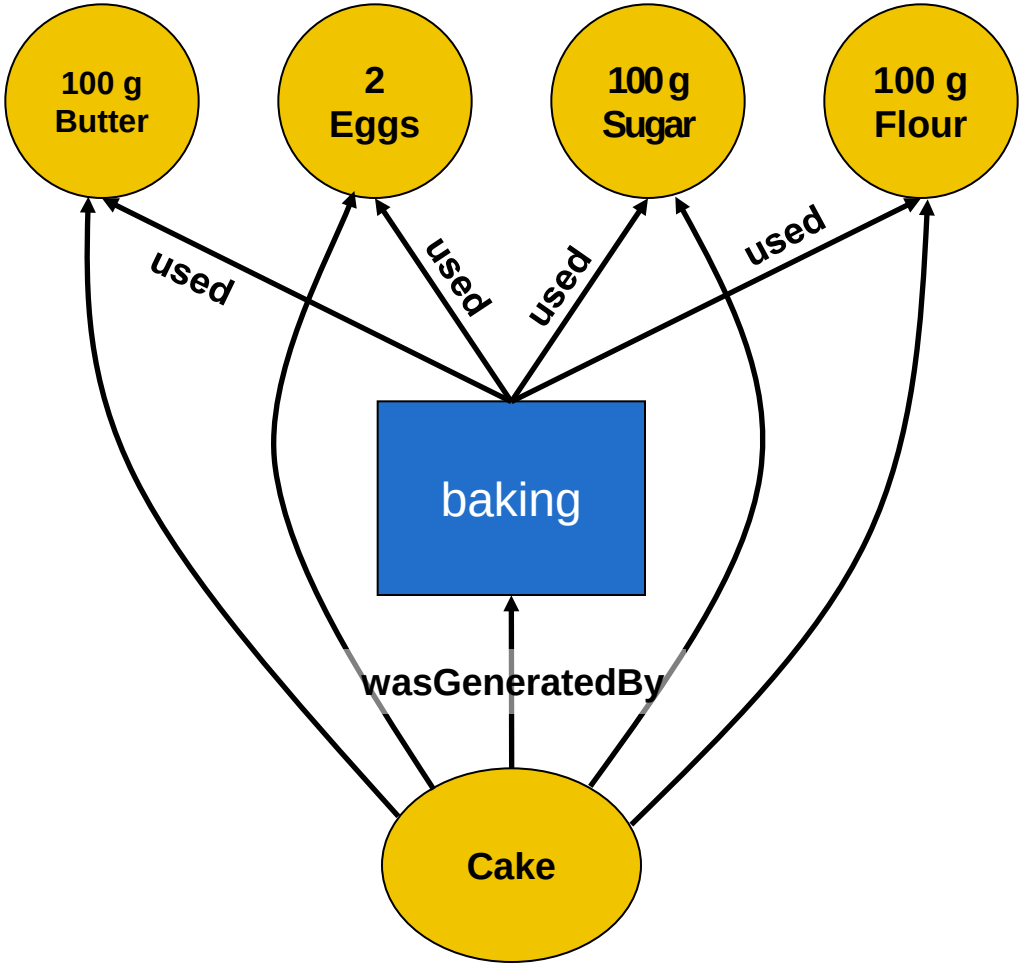
- Entity
- Activity
- Agent

Edges

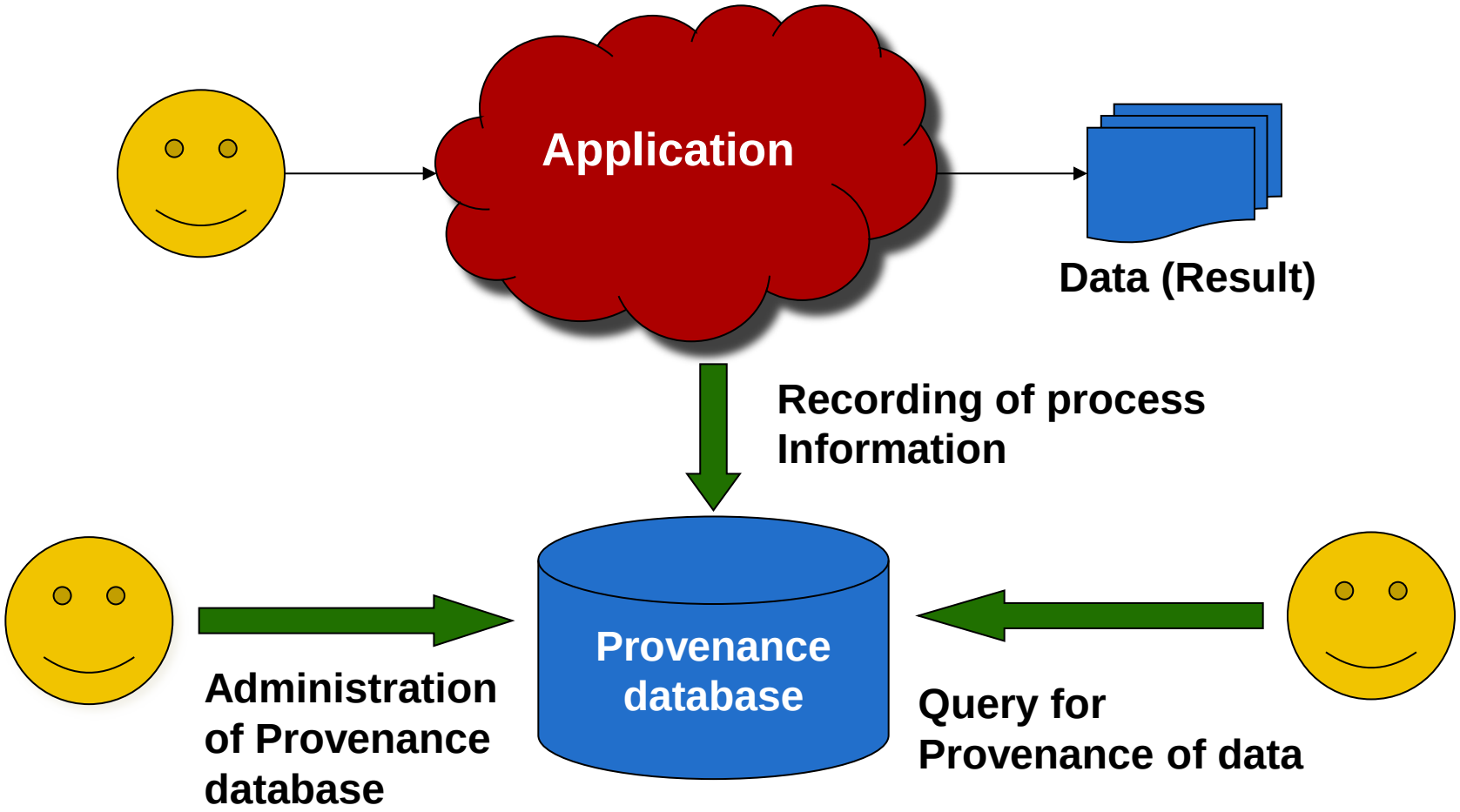
- association
- responsibility



Baking a Cake

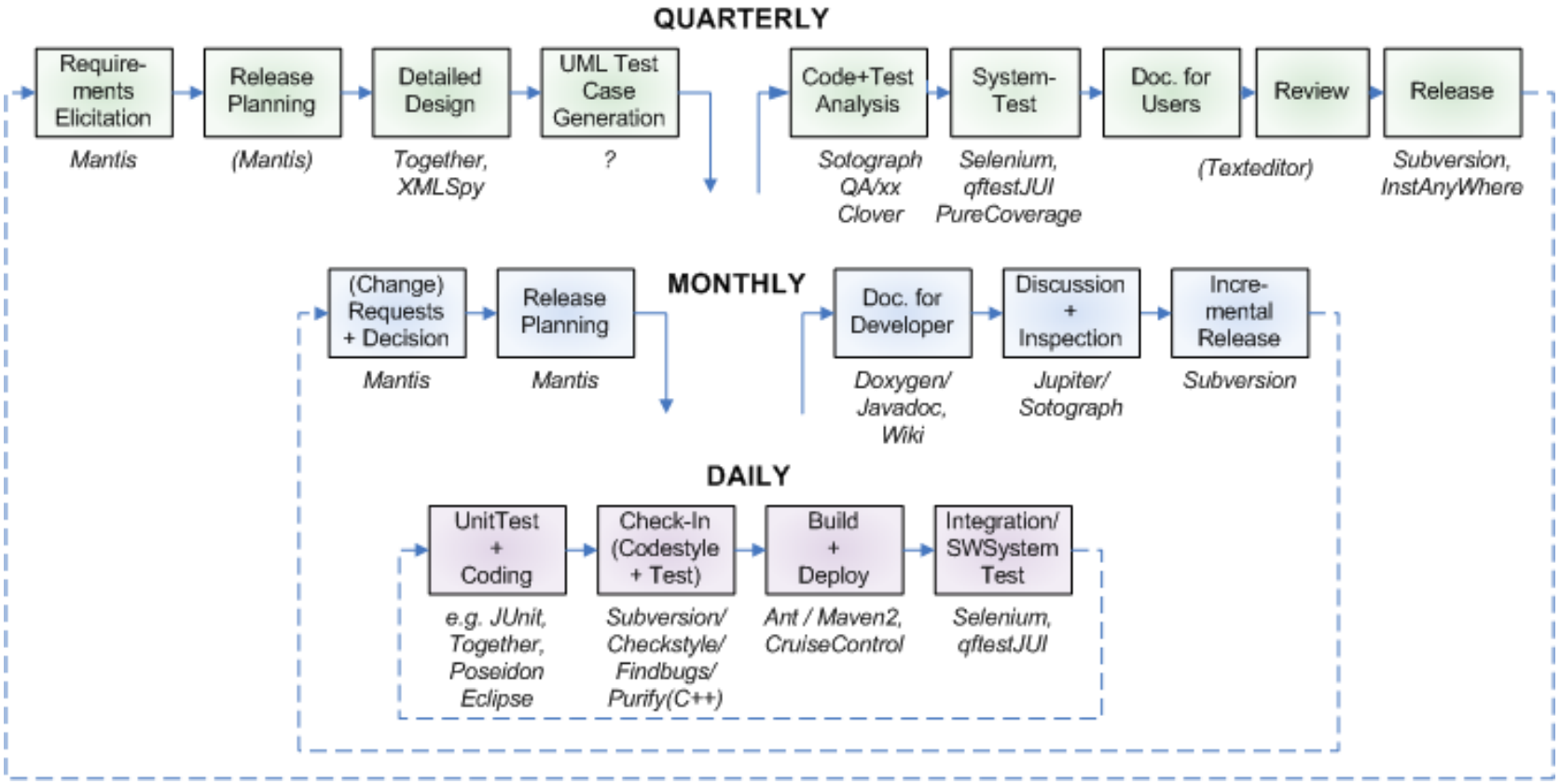


Provenance Life Cycle



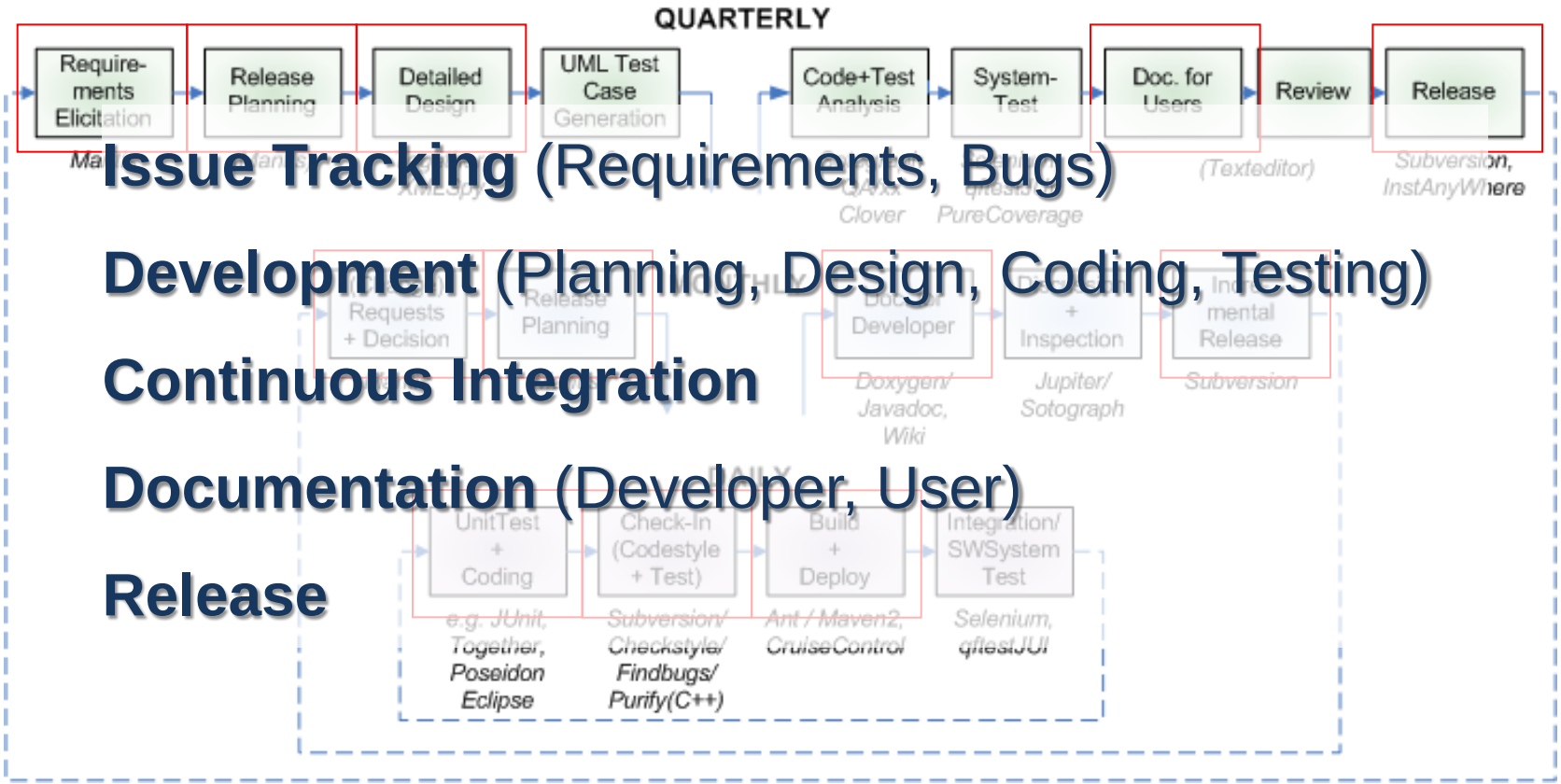
Software Development Processes

Typical DLR Software Development Process



DLR Software Projekt- und Entwicklerhandbuch, M. Bock, A. Hermann, T. Schlauch, 22.10.2009

Process Steps



Issue Tracking (Requirements, Bugs)

Development (Planning, Design, Coding, Testing)

Continuous Integration

Documentation (Developer, User)

Release

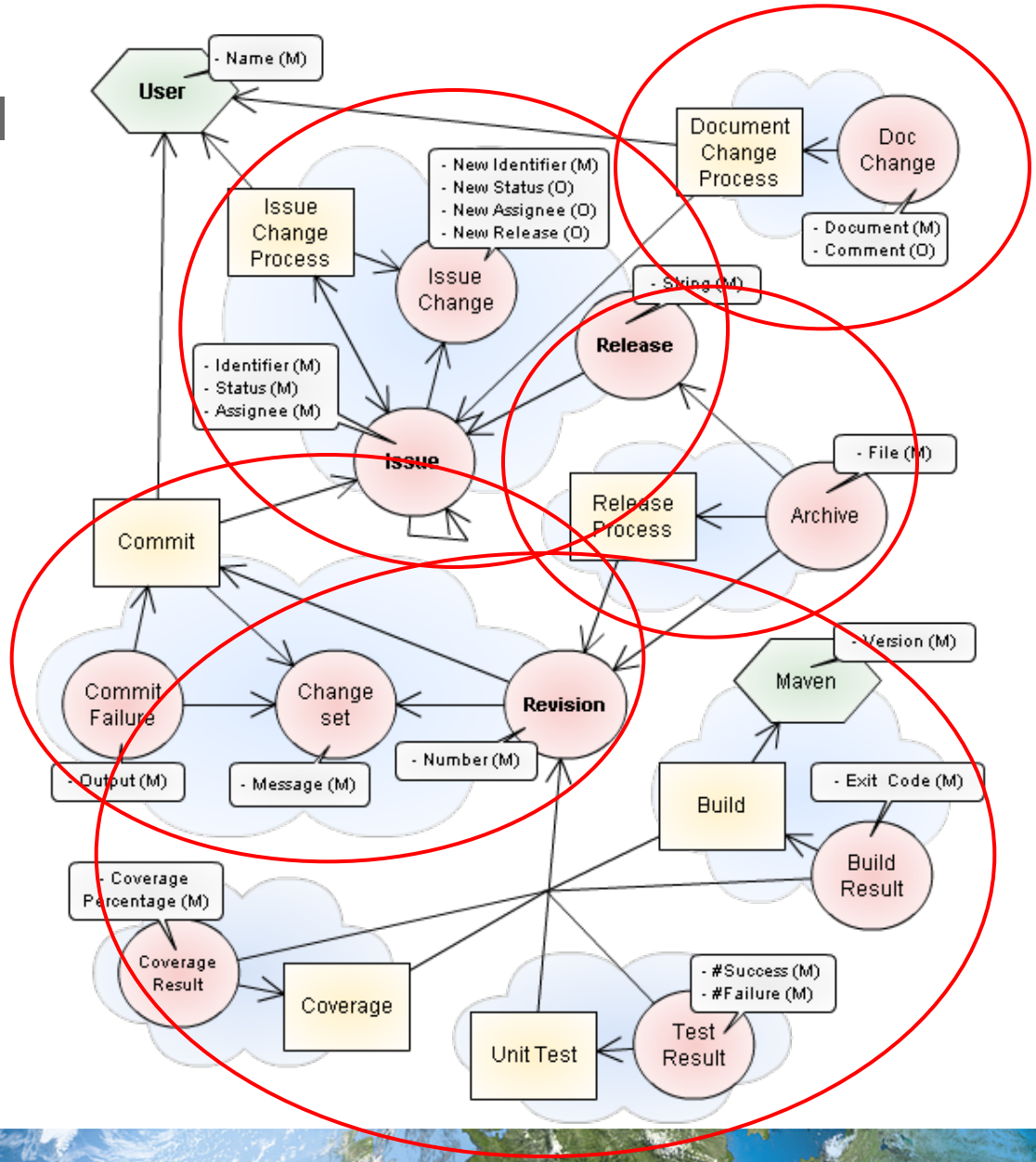
Provenance Model

Activities

- Issue Tracking
- Development
- Continuous Integration
- Documentation
- Release

Entities and Agents

- User
- Issue
- Revision
- Release



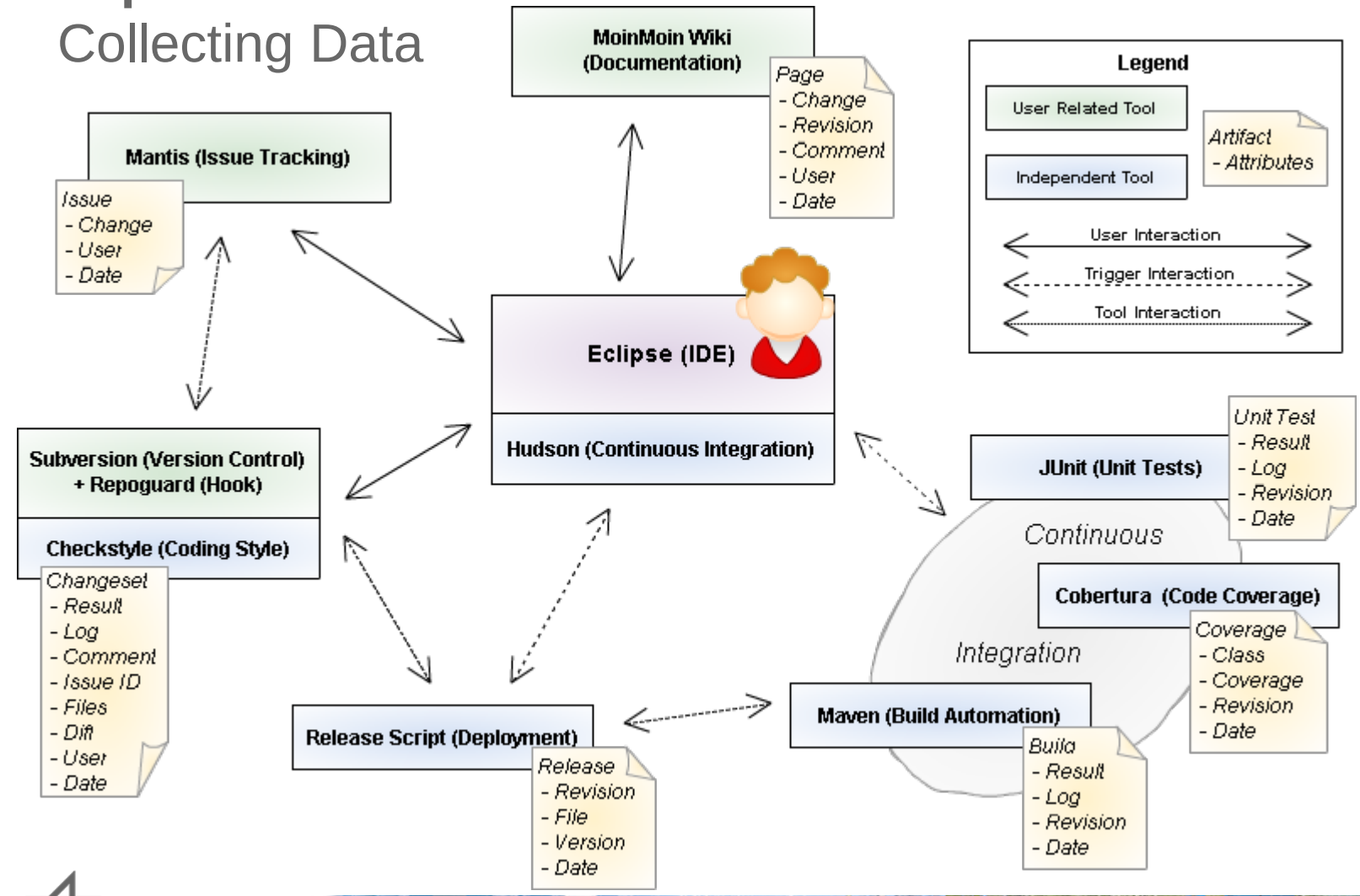
Questions and Problems

Error detection	*Which change set resulted in more failing unit tests?*
Quality assurance	*How many releases have been produced this year?*
Process validation	*From which revision was release X built?*
Monitoring	*How much time has been spent implementing issue X?*
Statistical analysis	*How many developers contributed to issue X?*
Developer rating	*Which developer is most active in contributing documentation?*
Information	*Which features are part of release X?*

Questions and Problems
Categorization

Single Tool	**Simple**	*What is the current overall code coverage?*
	Aggregated	*How did the number of unit tests change in the last month?*
Multi Tool	**Developer**	*How many issues were implemented by developer X for release Y?*
	Requirements	*How much time has been spent implementing issue X?*
	Errors	*Which requirement causes the most build failures?*

Implementation
Collecting Data



Implementation
Graph Database and Query Language

Graph Database *Neo4j*

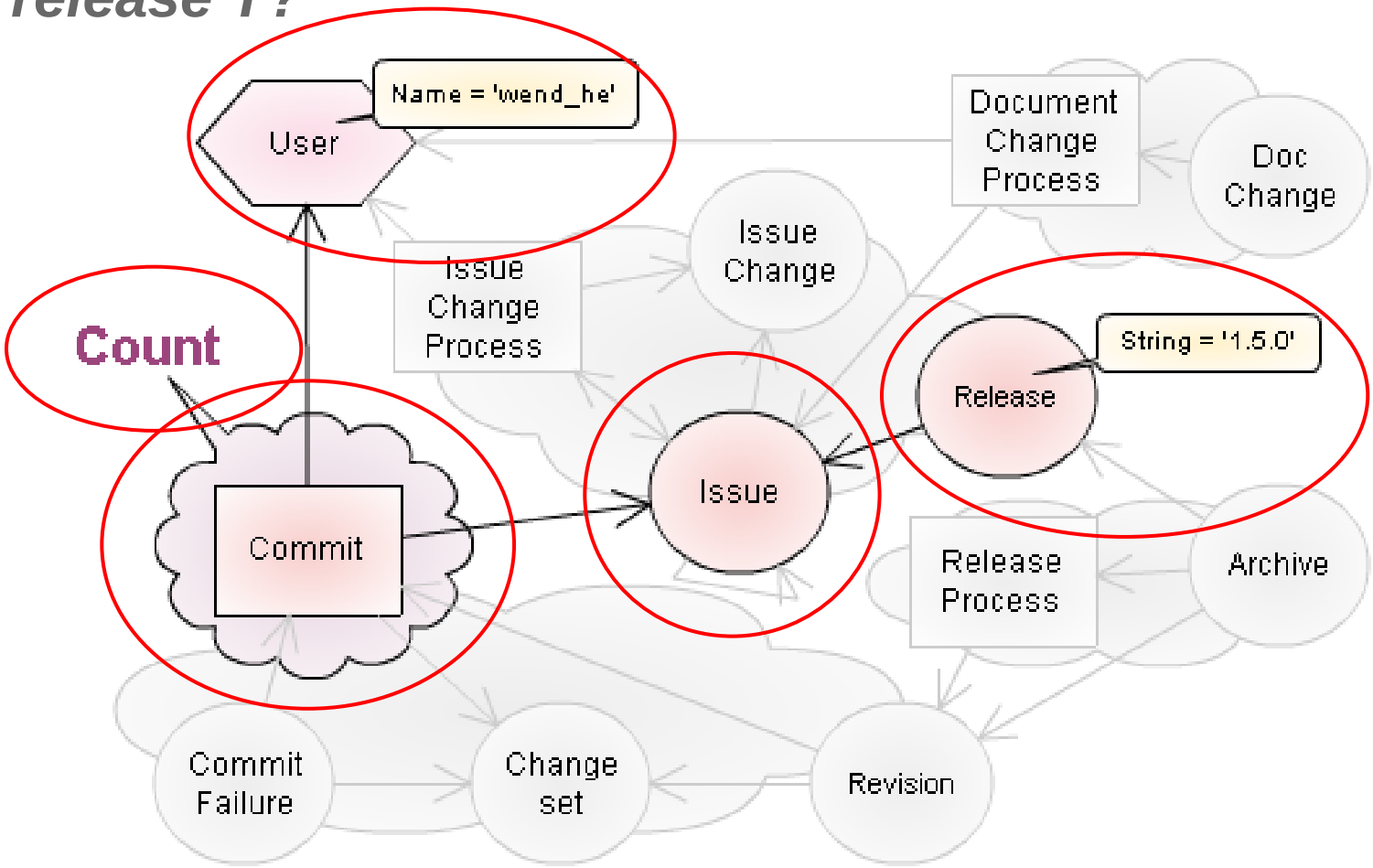
- High-performance NoSQL graph database

Query Language *Gremlin*

- Graph-based programming language for property graphs

Queries

How many commits did developer X contribute to release Y?



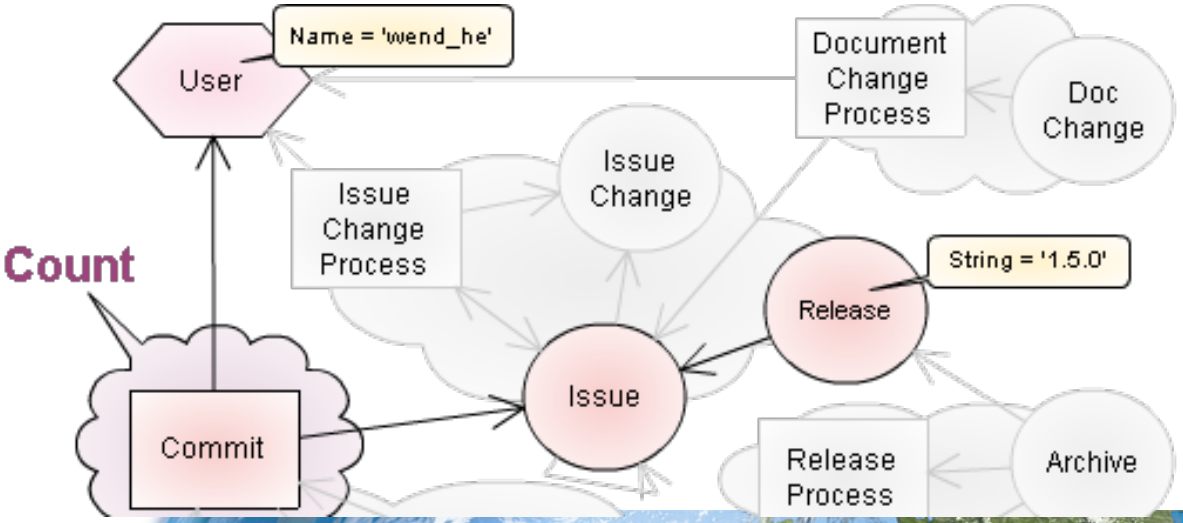
How many commits did developer X contribute to release Y?

```
$release := g:key($_g, 'string', string($release))

$commits := $release/outE/inV/inE/outV[@type='commit']

$relevant := $commits[outE/inV[@type='user' and
        @name=string($developer)]]

$count := count($relevant)
```



Which requirement causes the most build failures?

```
$ids := g:dedup(g:key($g, 'type', 'issue')/@identifier)
$results := g:map()
foreach $id in $ids
    $issues := g:key($g, 'identifier', string($id))
    $revision := $issues/inE/outV[@type='commit']
                 /inE/outV[@type='revision']
    $build := $revision/inE/outV[@type='build']
                 /inE/outV[@exit_code>0]
    g:assign($results, $id, count($build))
end

$most := g:keys(g:sort($results, 'value', true()))[1]
```

Open Research Topics

- Hiding the complexity of queries
- Visualization of query results
- Standardized semantics/ontology for software development processes

Questions?

Summary

- **Recording Provenance during run-time**
- **Deep insight into software dev. processes**
- **Higher trust in software quality**
- **Allows reuse with more confidence**
- ***Current research field!***

Andreas Schreiber
Twitter: @onyame
http://www.dlr.de/sc