AGILE DEVELOPMENT AND DEPENDENCY MANAGEMENT FOR INDUSTRIAL CONTROL SYSTEMS

Authors : B. Copy, M. Mettälä, CERN, Geneva, Switzerland

icalepcs 2011

Abstract

The production and exploitation of industrial control systems differ substantially from traditional information systems; this is in part due to constraints on the availability and change life-cycle of production systems, as well as their reliance on proprietary protocols and software packages with little support for open development standards [1]. The application of agile software development methods therefore represents a challenge which requires the adoption of existing change and build management tools and approaches that can help bridging the gap and reap the benefits of managed development when dealing with industrial control systems. This paper will consider how agile development tools such as Apache Maven for build management, Hudson for continuous integration or Sonatype Nexus for the operation of "definite media libraries" were leveraged to manage the development life-cyle of the CERN UAB framework [2], as well as other crucial building blocks of the CERN accelerator infrastructure, such as the CERN Common Middleware or the FESA project.

Objective

Have a global overview of your software engineering processes while relying on best-of-breed off-the-shelf products

Issue Management

This process focuses on collecting, prioritizing and refining customer demands and internal product quality feedback. Whether use cases, requests for new features or defects identified in existing products, such inputs must be classified and scheduled so as to reduce the risk of unwanted side effects and ensure timely release delivery. Change brought to a product must always take for reference a prior request for such change (once again, to reduce the risk of unidentified and unwanted modifications that could break a working product).

ATLASSIAN JIRA®

Change Management

This process ensures that changes can be audited, and grouped into coherent set of modifications. Sets of modifications will eventually compose a release. Change management is often seen as a burden by software developers, but becomes a key activity when it is coupled with release management – for instance, once a product version is out in the wild, it is essential to know exactly what differentiates this version from the one that worked better a few weeks or months ago.

SUBVERSION

Dependency Management

As agile teams happily release ever improving deliverables, being able to orchestrate dependencies among these deliverables becomes an increasingly difficult task. Establishing clear policies and allowing the definition of dependency ranges becomes a very important aspect of software project management. Considering that software project increasingly rely on third-party open source dependencies, it also becomes important to state and monitor the provenance of such dependencies. Sonatype Nexus is the Maven repository used to manage dependencies centrally.

Apache Maven Project http://maven.apache.org/ Nexus

SVN Commit Comments

SVN Version Tags

RDF W3C Semantic Web

RDF W3C Semantic Web

RDF W3C Semantic Web

Javascript Object Notation (JSON)

RDF

Resource Description Framework (RDF) is a standard model for data interchange on the Web. RDF extends the linking structure of the Web to use URIs to name the relationship between things as well as the two ends of the link (this is usually referred to as a "triple"). Using this simple model, it allows structured and semi-structured data to be mixed, exposed, and shared across different applications.
We perform transformation from proprietary data (e.g. JIRA issue entries) to standard RDF.

JSON

Javascript Object Notation (JSON) is used to translate RDF XML triples into a web browser-friendly notation.
All RDF data is merged into a large JSON object graph and fed to the Exhibit framework.

Simile Exhibit

The MIT Exhibit framework offers a generic way to represent, query, filter and visualize RDF data. We use the Exhibit Framework and have integrated a new visualization yet not supported by the framework for the representation of dependency trees.

Exhibit Framework

Visualizations

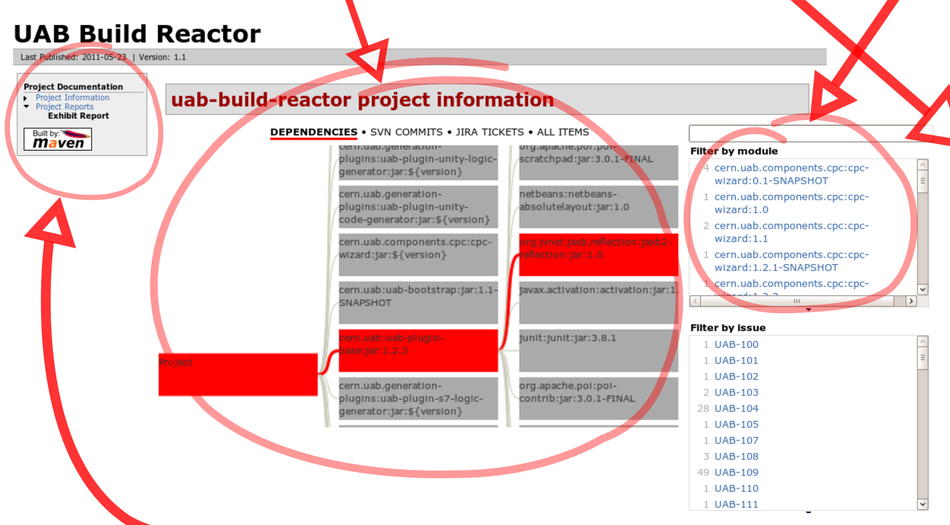
Data visualizations represent large amounts of information in an immediately understandable form. Maps, pie charts or tables are traditional visualizations, but others such as heat maps or sparklines can also deliver great expressiveness. In any case, visualizations are decoupled from the data they represent.
The Exhibit framework supports Timeline visualizations (middle), we have extended it with an interactive, Javascript, dependency graph visualization (left).

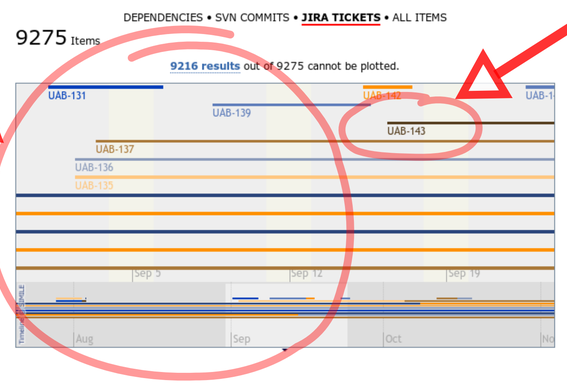
Queries and filters

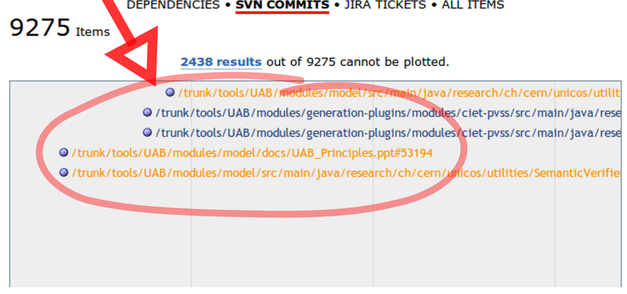
The Exhibit framework automatically generates filter and free-text search dialogs using the data at hand. Filters apply across all visualizations, ensuring view consistency.
In our example, it is possible to filter information per JIRA issue key (middle fig.), or per SVN revision number (right fig.).

Navigation

The Exhibit framework allows to navigate between RDF triples once the navigation paths have been configured. It is therefore possible to follow a dependency's version to the JIRA issues it was composed of, then on to the SVN commit operations performed in the context of one or more selected JIRA issues.







Build Integration

We wrapped data extractions and transformations into an Apache Maven site plugin – seamlessly integrating engineering process information into a web-based Maven documentation site [4].
This is useful for continuous integration software packages that support Maven but would not necessarily understand RDF or be able to handle the Exhibit framework.

Conclusions

Integrating software engineering process data sources is useful in order to obtain a global vision of development and release activities. While most engineering process deliver access to structured data, integrating and correlating this information is currently a non-trivial task.
While our usage of RDF as a common data format has certainly proven a workable approach, certain limitations subsist in the current support of RDF. The Exhibit framework for instance was designed highly interactve representation of low volumes of information.

A sample extraction of the UAB project activity over the course of 14 months, totalling about 600K lines of code, yielded a 5 megabytes Exhibit JSON input file containing 9275 records, which the Exhibit framework handles with difficulty (requiring a 20 seconds initial startup time and delivering occasional sluggish data filtering performances).
A complete rewrite of the Exhibit framework (Exhibit v3) is interestingly under way in order to ease the integration of third party visualizations and make it able to cope with datasets gathering more than 5 millions of records.

References

[1] R. Barillère, Ph. Gayet, "UNICOS A Framework to build industrylike control systems, principles and methodology", ICALEPCS 2005, Geneva, Switzerland, WE2.26I
[2] M. Dutour, "Software factory techniques applied to Process Control at CERN", ICALEPCS 2007, Knoxville Tennessee, USA, http://www.JACoW.org.
[3] APM Group, "What is ITIL ?", 20072011, http://www.itilofficialsite.com/AboutITIL/WhatisITIL.aspx
[4] Apache Maven Project, "Guide to creating a documentation site", http://maven.apache.org/guides/mini/guidesite.html
[5] B. Copy et al., "Model Oriented Application Generation for Industrial Control Systems", ICALEPCS 2011, Grenoble, France, WEAAULT02
[6] Massachusetts Institute of Technology (MIT), "The Exhibit Framework", 2011, http://www.similewidgets.org/exhibit3/

JACOW ID WEPKS001

Authors : B. Copy, M. Mettälä, CERN, Geneva, Switzerland

10 - 14 October 2011

CERN EN Engineering Department