

## Working more dynamically with Architecture

Stefan Dreverman

**During the past six months the architecture department of Portbase, a service provider in the Dutch harbor, has executed a pilot with an architecture repository to manage architecture principles. The tool that forms the basis for the repository is zAgile's Wikidsmart, a semantic plug-in for Confluence, the enterprise wiki from Atlassian. The result was that the quality of the architecture products has improved and the time-to-make was reduced. It has also enabled focused context searches, sharing with the organization and flexibility for future additions and improvements.**

### Time for a change

The team consisted of three architects and one lead architect. They identified a number of problem areas around efficiency and collaboration with respect to working with the architecture:

- Information and knowledge was widely dispersed in the company (in documents, in the wiki and primarily in the minds of employees)
- Decisions and motivations of architectural decisions were not documented
- It was unclear whether all concerns were properly addressed in the architecture
- It was impossible to validate the correctness of the applied architectural principles.

The past years there had been no time to make the necessary changes: the speed with which the development process was executed, as well as the limited resources of the team, left them no time to make improvements.

A window of opportunity arose after a merger. A new company-wide long-term vision had to be created. This requirement spawned little project start-ups. It provided an ideal moment to get up to speed: the tasks and responsibilities of the architecture department were defined in a more formal architecture process. After that, it was time for setting up the facilities for a repository.

### Back to basics

The architecture principles were defined first. The plan was to create a collection of basic principles, which would be complemented by context specific principles from every project that followed. In this way, the coverage of the documented architecture would naturally grow with every project.

Excel was used to document the first forty principles. At that point, it became hard to keep track of relations (e.g.: deduced from, supports, hinders, etc.) between the artifacts. Excel was not (by far) capable of supporting this. Similarly, the Confluence wiki, which was already used for documentation in the development process, was also not able to support this function in a satisfying way.

## Tooling: Web 3.0 meets Web 2.0

More was needed than a plain text editor or a spreadsheet. This started the search for a new tool. To keep the selection process pragmatic and lightweight, a few basic requirements were defined. This can, very briefly summarized, be formulated as: to easily store, link, share and search information about architecture principles<sup>[1]</sup>. We did not seek it out at the time, but the solution was to be a merger of Web 2.0 (collaboration) and Web 3.0 (semantic meaning) technologies.

A short online search showed various possible approaches, from modeling tools that share information (like BizzDesign Architect, PowerDesigner) to information sharing tools that can incorporate information models (Wikidsmart). The latter stood out because of its simplicity. Wikidsmart is a semantic plug-in for Confluence wiki: by adding semantics to a wiki, the structure and formality of an information model is combined with the freedom and informality of a wiki. It has room for structured information and for unstructured additions: all things that do not (yet) fit in the structure, can be added via the standard-wiki functionality. Recognizing the potential benefits and ability to overcome challenges outlined above, a pilot was started with Wikidsmart.

## A dynamic information model

To obtain structured information, an information model is needed. The architects defined the model in a joint effort. They decided which information was initially incorporated in the model. The pilot used a model which incorporated principles, stakeholders, requirements, projects, systems and quality aspects.

Artifacts were then linked to each other by defining triples<sup>1</sup>. The following list describes some of the relations that were defined:

- "stakeholder" has "requirement"
- "requirement" supports "architecture principle"
- "project principle" deduced from "architecture principle"
- "project" has "project principle"

These definitions create possibility for Portbase to link stakeholders to a project via project principles and architecture principles. It is a common practice to define stakeholders for a project. However, some of the project principles can introduce stakeholders which are *indirectly* linked to the project. By re-using an existing architecture principle, all stakeholders of the requirements that supports that architecture principle, are indirectly introduced into the project. Connections like these already existed in projects, but it was nearly impossible to document and track them.

The process of defining the model was the most difficult stage in the project. In most applications the information model is fixed, but now we had the opportunity to make one that exactly fit the needs of Portbase. There was much debate about how the model should look. For every version there were several pros and cons, but no clear winner. After several iterations, the effort to improve was became too big for the quality that could be gained and thus the model was implemented.

## The power of relations

Soon after the team started working with the model, the power of the semantics was not simply in expressing the information itself. Rather, it was the possibility to extract information. Queries proved to be a powerful way to conduct impact analysis like: "Which stakeholder has the most requirements with the quality attribute "security", which architectural principles are related to those requirements and what parts of the system have to follow those principles?".

---

<sup>1</sup> Wikidsmart uses the Resource Description Format (a W3C standard) to describe information models: <http://www.w3.org/TR/rdf-concepts/>

Or, if those examples are too complex, queries like "Which quality aspects are most important for a stakeholder?" also proved to provide valuable information. By visualizing these relations a meaningful dialog can be started between various stakeholders.

## **A more dynamic Project Start Architecture**

Somewhat bigger analysis were also standardized using the semantic plug-in. A template was made to create a Project Start Architecture (PSA). For projects, the architectural impact of each requirement was put into the model by using their instances and relations.

Parts of the PSA are filled by a number of queries. They are used to retrieve information about stakeholders, principles and decisions, covering more than fifty percent of the PSA. The remaining parts (business & architecture drivers and project transcendental issues) still had to be entered in plain-text.

Two architects have defined PSA's for three projects in only one week, where the plain-text PSA typically took one architect six to seven days to complete. Time is gained here by only entering the information once. Since the bulk of the information that is needed for the PSA has to come from the model, there was no need to browse through large documents to find the necessary information.

Furthermore a higher quality PSA is delivered: It does not only present the right (amount of) information about a project. Background information is also provided when needed. Since all the information is linked, the reader can "zoom in" on an artifact and get more information about the why, what and how. Instantly, without having to search for the necessary documents, pages and paragraphs.

## **Connecting development tools**

Although the model in the pilot is focused on information about architecture principles, it can be extended to cover other areas of the software development process. zAgile, creator of Wikidsmart, has spotted more possibilities for both information models and integration and is moving forward rapidly. A few months after starting the pilot, a commercial version 1.1 was presented (and while I'm writing this article, a version 1.2 has been released), which contains full integration with JIRA, an issue tracking application. Projects, issues, users, etc. from JIRA can directly be used within Confluence and within the information model. This enables definition of relations between JIRA and Wikidsmart content and executing queries. And it doesn't stop there, because zAgile's next target is to integrate with SVN and CruiseControl.

The gain in integrating domains in the software development process is not found in one particular domain. While every domain, like architecture, remains working with their own information, integration provides a platform to improve traceability quality in the process as a whole.

## **Conclusions**

For Portbase, the results of the pilot were presented to the departments product development, technical application support and test. Since the audience was mostly IT, the bulk of questions were not about what they could do with it, but about how it actually worked. However, some employees spotted great potential in fine-tuning and integrating the flow of information to improve the quality and speed of software releases. At this moment in time, some of those possibilities are being investigated.

For the field of IT Architecture, information is becoming increasingly accessible and it can be modeled to the needs of the user, too. This creates the possibility for architects to dynamically mold (the flow of) information needed for the architecture process. By using semantic tooling, the field of IT architecture can create opportunities to integrate information with other roles in the software development process. This represents a bold new world for the enterprise. Now the IT architecture no longer has to be a collection of disparate "black boxes" but rather completely integrated, transparent, and connected to the business operations.

Stefan Dreverman

**Sogeti Nederland BV**

[Stefan.dreverman@sogeti.nl](mailto:Stefan.dreverman@sogeti.nl)

---

[1] Farenhorst, R., Lago, P., van Vliet, H.: Effective Tool Support for Architectural Knowledge Sharing