Architecture Deployment.

White Paper Resulting from Architecture Forum Meeting

November 2-3, 2010, Philips Healthcare, Best, the Netherlands

Edited by:

Dr. Gerrit Muller, Buskerud University College, Embedded Systems Institute

Mr. Peter Korfiatis, Stevens Institute of Technology

Input was provided by the following participants in the Architecture Forum:

Name	**Organization**
Leandre Adifon	OTIS (by phone)
Frank Benschop	Philips Healthcare
Kees Kooijman	FEI Company
Peter Korfiatis	Stevens Institute of Technology
Pierre vd Laar	ESI
Bjørn Victor Larsen	Kongsberg Defense

Name	**Organization**
Phil van Liere	Philips Healthcare
Sjir van Loo	ESI
Gerrit Muller	Buskerud University College/ESI
Sølve Raaen	Kongsberg Maritime
Nico van Rooijen	Philips Healthcare
Martin Simons	Daimler

Published on March, 2012

1. Introduction

Many architects struggle with their role in deploying an architecture. Creating and describing an architecture can be a “paper” exercise; the real challenge is to create a realization that complies with the architecture. In the preparation of this forum meeting we formulated the following questions:

1. How to enforce an architecture?
2. How does the architect do this without formal power?
3. Does an architect need formal power?
4. Is enforcement needed?
5. How to get architecture as intended?
6. How to share common view?

Some other quotes from the preparing discussion:

- What is "our" (architects’) boundary?
- We want people to use it (the architecture).
- What does an architect need for successful deployment?
- Architects build acceptance during creation of the architecture.

2. The Architect’s Role

Architecture deployment is closely related to the vision on the role of that architect: what do architects do, how do they architect, what are the responsibilities, how are they empowered? When asking architects themselves, there is a large probability that the answer is “that depends”, or “it is a balancing act”. Bredemeyer and Malan give a clear description of the architects’ role in [Bredemeyer 2006] they describe the architect as working via influence rather than formal power. Bredemeyer uses a simple framework (what you *know*, what you *do*, what you *are*) to look at different levels of architecting (technology, business strategy, organizational politics, consulting, and leadership). This description makes it clear that architects work in a complex environment, full of tensions. This explains why the typical architect answers, “that depends” and “it is a balancing act”.

Michael Duijvestijn from Philips gave an introductory presentation. He outlined the process that the MRI business unit was going through to manage and architect their product portfolio. This process was facilitated by PRTM, a well-known project management consultancy firm. Figure 1 shows the strategy process flow. Capability management is at the core of this approach. The capabilities are determined both top-down (what opportunities do we want to address given market trends and needs) as well as bottom-up (where should the system evolve into, based on technology innovations and architectural improvements). This is transformed into an integral roadmap with a 10 year view. The roadmap is used to define a project portfolio with a 2 year view.

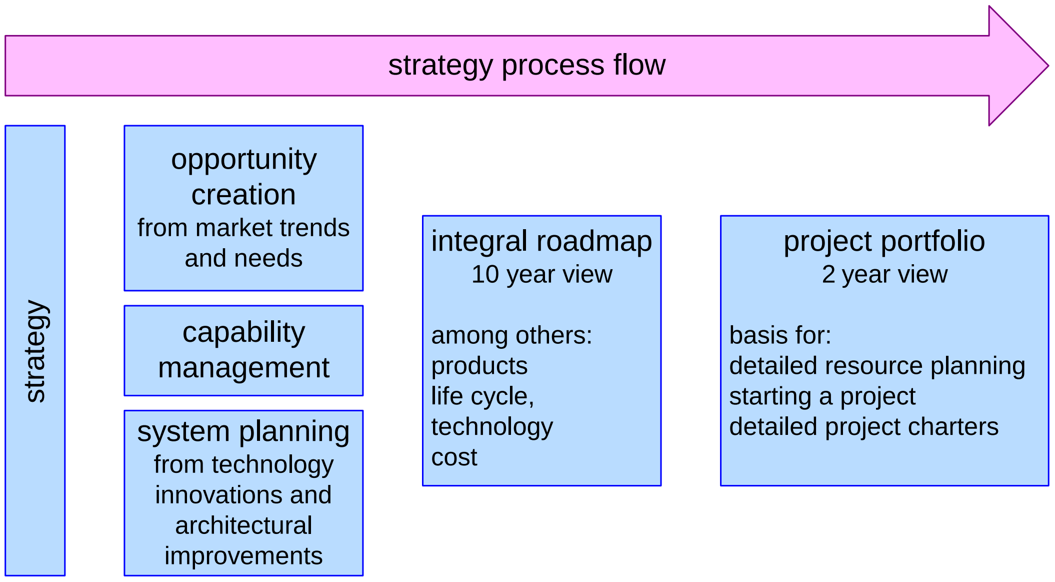

Figure 1. Strategy process flow as deployed by Philips MRI business.

The strategy process is embedded in the organization by defining a number of “cultural pillars”:

- Fact-based pro-activity
- Disciplined creativity
- Passionate collaboration

These cultural pillars are of special interest to architecture deployment and the architect's role. The interesting aspect of these pillars is that they capture or express tensions. *Pro-activity* is desirable, but often handicapped by uncertainties and unknowns; the wish for *fact-based* is understandable, but far from trivial. "Disciplined creativity" sounds as a contradiction in itself; innovation needs *creativity*, the business needs *discipline*; how to combine these two? The forum did discuss this tension in March 2008; see [Muller 2008]. Finally, passionate collaboration is a fully understandable need for an organization that is globally distributed and has a complex network of suppliers and partners. At the same time, we should realize that individuals and local groups have private concerns that do not always align with the global direction. How do they collaborate passionately when it threatens personal or local interests?

Fact-Based pro-activity. Which facts really matter for your business? Developers should use past experience to gather these facts and link them to specific user scenarios. A prerequisite is agreement on metrics. The challenge is to match objectives (what do we want to achieve) with pragmatics (what do we know, what can we measure). Ill-willing persons have room for manipulation that is camouflaged by the objectiveness of a fact-based approach.

Pro-activity might be triggered by feelings of elegance, or the intuition that it can be wrong. These modes of operation are only allowed when supported by facts. Many organizations improve when it really hurts; that is reactive instead of pro-active. Pro-activity apparently needs some trigger, like a vision (for instance, articulated as roadmap) or operational understanding and foresight (e.g. the margin is reducing 10% per year, we need to increase the margin before we hit the wall).

Disciplined creativity. This tension can be mapped on the time-horizon: short-term freedom to act, long-term freedom to think. The discipline part requires well-defined boundaries that are communicated. The boundaries are often incomplete. Lack of capturing the boundaries can make them difficult to communicate.

Passionate collaboration requires clear and common incentives that cope with short-term versus long-term tensions. Without incentives the collaboration might turn into passionate negotiations. An interesting convention to enforce this thinking is "**them** is only used for competitors”; all others, such as partners, colleagues from other departments or overseas, customers, and suppliers belong to “us”.

When comparing their own companies to this strategy flow, some participants indicated that their company was not far along in terms of integral thinking. For example, some companies maintain a strong cost reduction focus with separate budgets for development and service, or a product roadmap that is not linked to market, technology and competencies. The consequence is that these companies are reactive rather than pro-active; for example, obsolescence can be seen as driver to change.

United Technologies (UTC) presented another successful deployment of an architecture at Otis elevators. A small team of architects obtained support from upper management and came up with a functional architecture that was mapped to the elevator’s physical parts, as well as the needs of later life cycle activities such as logistics, manufacturing, and service. In this case, market pressure and competitiveness were triggers, and leadership commitment, a solid business case (mostly cost reduction, some value creation), and availability of the right architecting resources enabled successful deployment. The architects needed to educate the company champion and others in functional decomposition to get away from traditional physical partitioning.

3. Cases

Two members presented a case of how architecture is being deployed in their business unit. One case is in a very mature mechanically dominated business. The other case is a complex system with software that has evolved over three decades.

The mechanical dominated business case.

The engineers and most stakeholders were completely thinking in the existing physical decomposition as it had evolved over decades. This pure physical thinking blocks potential improvements and evolutions.

The chief architect took the organization through the following process:

- Create a functional model of the system, starting with operational scenarios and refining it in main and leaf functions.

- Synthesize a functional segmentation.
- Rethink the physical partitioning by allocating functions to minimize coupling and maximize independence of testing.

The result of this process was that the step up in abstraction to functions allowed the engineers to come up with new architectural solutions with improved performance and cost. The final architecture was more modular with more standardized interfaces, facilitating global development and production. With increased market and time pressure, these benefits created a significant competitive advantage.

The whole process was far from trivial, since this kind of change induces a lot of resistance. The whole process took ten years. Particular factors that made the architecture evolution possible are:

- the presence of a company champion that supported the change ("who puts his ass on the line")
- the allocation of dedicated resources (although quite limited in numbers)
- the belief that there is a solid business case
- the presence of a persistent and persuasive architect

The software intensive system.

This system has evolved over 25 years into a system with 8 Million lines of code in 600 building blocks. The software in this system suffered from scattered ownership, a lack of interface management, and a lack of overview. This awareness triggered an architecture evolution, where the software architecture was transformed in three phases.

1. Re-factor in well separated functional clusters
2. Facilitate independent deployment of functional clusters
3. Harvest by independent development of functional clusters

The current status after Phase 1 is a refined allocation of all building blocks to about 20 functional clusters. These functional clusters form the foundation for the organizational allocation. Every functional cluster has a cluster owner and an architect, and every cluster is allocated to one of the eight group leaders. The interfaces between the clusters have been analyzed further and improvements are identified to decrease coupling. Currently, every functional cluster is working on an architecture overview A3 as proposed by Borches and Bonnema [Borches 2010] to recover the overview of the entire software architecture.

The main benefits of the architecture evolution are that a shared view and vocabulary has been created, and that the functional cluster structure provides a means for organization, ownership, archive structure, documentation and interfaces. The functional view is not "pure" however, it includes the main concerns in sufficient detail to engage the designers. The impurity is that it mixes several views with functionality as the primary driver, but partitioning, layering, and physical decomposition are also somewhat included. The deployment idea behind this approach is that recognition and engagement of the designers is more important than the purity of the representation.

4. How to Enforce an Architecture?

The word enforcement triggers quite some emotion. Several participants said "let's get rid of the word enforcement." The risk of enforcement is that trust is eliminated and that collaboration is driven by force rather than motivation. Enforcers can easily lose their feedback channels.

There is a clear cultural component in this discussion. In the Dutch and Norwegian cultures enforcement is very difficult. In these cultures the preferred view is that architects help to solve problems. Enforcement becomes an issue when an architecture is not meeting the need of its stakeholders, which are often conflicting. Should architects enforce in such a case or does it fall to management? And how do architects cope with stakeholders that cannot be involved at the right time? At small scale, e.g. at designer level, enforcement can work in these cultures.

Several enforcement models were discussed:

- Empowerment by management
- Authority natural and earned

- Enforcement by process
- Persuasion
- Operating on a shared vision

These different models triggered a heated discussion about consensus. This word is typically a cultural word again. In consensus-based operation, stakeholders keep talking and aligning until a direction is chosen that is accepted by all. The result of consensus can be that the chosen solution is a compromise between all interests. This mode of operation is also known under the terms committee design or polder model. Another interpretation of consensus is that stakeholders knowingly disagree but accept and commit to a solution anyway. One risk of consensus building is that stakeholders escape into descriptions that are too abstract.

Persuasion is a model where the architect builds relationships and a position of authority. Persuasion requires clear communication. However, even when the architecture is clearly captured in diagrams and the designers have committed themselves, the outcome may not be as intended. In other words, initiating an architecture is insufficient to enforce it. Monitoring and adaptation of architecture or design is also necessary.

To have influence on the deployment of an architecture, architects should be involved in a number of key processes, such as specification, design reviews and change control. The process embedding can take place by membership in boards, e.g. an architecture board, change management board, etc.

Operating on a shared vision also triggered some discussion: can it exist at all, at what abstraction level? Very abstract shared visions are seen as useless while more detailed elaborations might contain too much information for many stakeholders.

5. Deployment Impact Factors

The discussion about deployment models and the presented cases show that there are many factors that impact the effectiveness of deployment:

- Human
    - Psychological (motivation, trust, status)

  - Social (local interests)
  - Political (power)
  - Cultural
- Time
  - Stakeholders are out of phase
  - Long-term insight (think) versus Short-term commitment (act)
- Process, governance
  - Naturally embedded
- Content
  - Right idea

The *human* context plays a dominant role in the deployment approach with both individual properties (psychological) and group properties (social, political, and cultural). Architects need awareness of these factors and skills to cope with them. The *time* dimension has impact because different stakeholders have different perspectives; they see different phases of the life cycle and experience different benefits and problems over time. There may also be a cultural difference in time orientation: long-term versus short-term orientation, see Geert Hofstede (http://geerthofstede.nl/culture.aspx).

The embedding and governance of architecting in a “natural” way will support architects in the deployment. Natural means here that it fits in the architectural way of thinking and doing. Architects inherently cope with lots of uncertainties, unknowns and counteract consequences of organizational or technical boundaries. Conventional processes, such as waterfall process and work breakdowns may conflict with these particular architectural needs.

The last impact factor is the content itself; do we understand business and life cycle, do we have the proper problems and needs identified, a sensible specification, appropriate concepts, and fitting technology?

We have combined these insights in the following principle:

Principle 11.1 Successful deployment requires an architect with:

- *the "right" socio-political behavior*
- *the ability to cope with challenges induced by time aspect*
- *supported by the "right" process and governance*
- *a fit-for-purpose architecture (it solves the problem)*.

6. How to get an architecture adopted and deployed?

We challenged all architects that were present with the following questions:

- What do you do to get an architecture adopted and deployed?
- What do you do to improve an architecture?
- What are your main success factors and road blocks?

From this discussion we collected the following set of common success factors for deployment (Appendix 1 gives for every category some quoted some individual statements):

- Involve stakeholders
- Secure management support
- Get facts (measure, model, test)
- Communication platform (meetings, workshops and infrastructure)
- Create understanding, simplify
- Create proper match ownership
- Develop a proper architecture, who's deficiencies are known, managed and accepted for the time being
- Manage short term, long term, roadmaps and planning
- Provoke out of phase stakeholders with specific proposals

Similarly, we collected a common set of roadblocks:

- Time pressure
- Lack of Key Resources
- Current project commitments get in the way
- Lack of Management Support
- Organizational resistance

Note the importance of the socio-political behavior and skills as mentioned in principle 11.1 in view of the common roadblocks. This is for many architects a struggle: their interest in the content (needs, concepts, technology, etc.), while their job requires socio-political behavior and skills. How to balance the socio-political world with the content world?

Summary and Conclusions

We can now revisit the questions that we formulated in the beginning:

1. How to enforce an architecture?
2. How does the architect do this without formal power?
3. Does an architect need formal power?
4. Is enforcement needed?
5. How to get architecture as intended?
6. How to share common view?

The goal is not to enforce an architecture, but to realize an architecture as intended. We agree that personal authority and persuasion, a “natural” embedding in processes, and working towards a shared vision are complementary and beneficial. The amount of empowerment triggers a heated debate, where in general architects are seen as working through influence rather than power. The sharing of a common view also results in a heated debate with as extremes consensus via a committee design (too compromised) and the visionary empowered shaper. A more fundamental question was raised whether a common view really can be shared, a sobering question.

The preparation triggered some other quotes to be revisited:

- What is "our" (architects') boundary?

This has not been discussed explicitly. However, implicitly, a rather boundary-less architect emerges. Or, alternatively, an architect mostly hits circumstantial boundaries: issues out of the influence scope of the architect.

- We want people to use it (the architecture).

Goals are more important than the means. Architects want the "right" result, the "proper" solution. Enforcement is not a goal, but in best case a means.

- What does architect need for successful deployment?

See Principle 11.1: Successful deployment requires: architect with "right" socio-political behavior, coping with time aspect, supported by "right" process, and the content is OK (architecture solves the problem).

- Architects build acceptance during creation of the architecture.

This statement matches with the discussion, see also some of the quotes in Appendix 1.

A main discussion in this paper is: Do architects ***enforce*** architectures? The general answer tends to: no, good architects achieve the intended architecture through influence, e.g. by using natural authority and persuasion, supported by "natural" processes. The consequence is that architects need significant socio-political skills.

Appendix 1, Some individual responses to success factors for deployment:

- Involve stakeholders
    - Early buy-in by all relevant stakeholders while the architecture is being developed
    - Make lots of drawings and presentations

  - Continually talk with key stakeholders
  - Participate in function allocation in the different teams
- Secure management support
  - Lobbying – finding friends rather than facts
  - Management sign off
- Get facts (measure, model, test)
  - Quantify behavior – measure and characterize the system
  - Complimented by small test and simulations
  - Modeling of architecture, model based assessments of performance, key performance metrics, fact based that support tradeoffs
- Communication platform (meetings, workshops and infrastructure)
  - Create a stronger communication platform, e.g. Software Architecture Team
  - Open communication – web portals – communication strategy
- Create understanding, simplify
  - Make current architecture understandable (create some order in chaos)
  - Ability to describe architecture in simple terms that all stakeholders can understand
  - Getting the magic out, making systems and explanations simpler
- Create proper ownership
  - For instance as described in the software intensive system
- Proper architecture, who's deficiencies are known, managed and accepted for the time being
  - Move functionality to create less dependency

- Synchronize developments in affected subsystems
  - Introduction of capability interfaces
  - Stabilize interfaces
- Manage short term, long term, roadmaps and planning
  - Establish a roadmap
  - Sustained satisfaction with solutions (>4 years)
- Provoke out of phase stakeholders with specific proposals

Literature

[Borches 2010] P. Daniel Borches and G. Maarten Bonnema, A3 Architecture Overviews; Focusing architectural knowledge to support evolution of complex systems, proceedings of INCOSE 2010 symposium.

[Bredemeyer 2006] Bredemeyer, D. and Malan, R., The Role of the Architect, www.bredemeyer.com/pdf_files/role.pdf retrieved August 9, 2011.

[Muller 2008] Edited by: Gerrit Muller, and Eirik Hole, Innovation, Disruptive Change, and Architecting. White Paper Resulting from Architecture Forum Meeting March 5, 6, 2008 (Dallas, Texas, USA)
http://www.architectingforum.org/whitepapers/SAF_WhitePaper_2008_6.pdf
retrieved August 9, 2011