Bridging the gap between an enterprise and its projects: Using the Zachman Framework to define IT Architecture

KnowFuture - Event
Bridging the gap

Agenda

- Introduction: we need Architecture Management
- The context
- Architecture Management@Swisscom Mobile
- The approach
- Some practical experiences
- Conclusions
Our Main Message

Architecture Management with the Zachman Framework:

- bridges the gap between an enterprise and its projects
- supports management of complexity (when you have to make several packages (and/or in-house developed applications) working together)

Agenda

- Introduction: we need Architecture Management
  - The context
  - Architecture Management@Swisscom Mobile
  - The approach
  - Some practical experiences
  - Conclusions
Do you know the spaghetti syndrome?

The story starts like this....

Day 1

I have a great idea for a new customer product.

That will be a very great and advantageous product. We want to launch it in 2 weeks!!

We have 2 weeks, let's build a product-specific solution...
That will be a very
great and
advantageous
product. We want to
launch it in 2 weeks!!

We have 2
weeks, let's build
a product-specific
solution...

I have a great
idea for a new
customer
product

Day 2

...And continues again... and again...

Day n

Day 1
...And finally: The technical landscape is like a big spaghetti plate!

- Applications were built one-at-a-time for individual products.
- Each application was its own center of the universe; interfacing was afterthought.
- Each application’s architecture has its own presentation, logic, and data layers.
- Many applications have their own security models and logins.
- Files and databases are islands and often store redundant data that is already stored in other files and databases.

...etc., etc.

At each time we have to make a change we have to touch everything and we need a lot of time and resources!

In other words:

- Projects work under high pressure; the product is for next week.
- Projects are focused in meeting deadlines, which is good by the way...

- This create an unavoidable trend to build solutions that are project-specific: "vertical solutions".
- At a point of time, the technical landscape is like a big spaghetti plate... and nobody takes care about the end-to-end architecture!
But, in the meantime, everybody wants:
- Reduced time-to-market
- Alignment
- Integration
- Flexibility
- Interoperability
- Quality
- Seamlessness
- Adaptability
- User-friendliness
- Usability
- Reusability
- Security

How can we get this from spaghetti?

Solution?

“We need a managed Architecture”

In other words:
- We need an Architecture
- We need to manage it
Agenda

- Introduction : we need Architecture Management
- The context
  - Architecture Management@Swisscom Mobile
  - The approach
  - Some practical experiences
  - Conclusions

Swisscom Mobile
- The leading mobile phone operator in Switzerland
- 66% market share with 3.675 M customers and 2600 points of sales
- 2002 revenue: 4112 Mio CHF
- 2400 employees
- 75% Swisscom AG, 25% Vodafone Group

The Swisscom Group
- Switzerland’s leading telecom company
- Positioned as the leading provider of mobile and fixed voice and data services and Internet-based services.
- Offers a comprehensive range of telecom products and services
- 2002 revenue: 14,52 Billion CHF
- 20,470 employees
What does Swisscom Mobile provide?

- Phone calls, short message service (SMS), voice mail
- Add-on services:
  - roaming (phone calls when you are in other countries), fax and e-mails (unified messaging), call forwarding, teleconferencing...
  - Mobile phone (micro)payment
- Third party content: railway timetables, stock prices, weather, bank account information, portal of services
- Business services: Corporate Mobile Network, Mobile Office, Machine-to-machine mobile data communication
- Public Wireless LAN
- Multimedia messaging, WAP surfing
- Games – downloadable and on-line
- Prepaid and post paid billing
- ... and more

Functional layers

**End-user Applications**
- Multimedia messaging, WAP portals, Games, Chat
- Third party information services, Corporate Mobile Office...

**Delivery Services**
- Rendering, Personalization, Service creation environment...

**Enabling Services**
- User identification, Authentication, Real-time billing interface, network gateways...

**Network**
- Circuit switches, Packet switches (GPRS), Public Wireless LAN, UMTS...

**Business Support**
- CRM, Billing, Provisioning, Finance, Supply Chain, Product Management...
Strategic Issues

- For a mobile phone company, technology is not business support or a business enabler - it is the business
- Telecom technology vendors have a dominant role in the industry
- Core technology and services are very standardised and well-served by package vendors
- Competition is in emergent value-added services:
  - It isn't feasible to develop in-house what you need to build them - you base them on packages
  - Development is mostly about configuring packages and making them work together.

Strategic Impact of Packaged Solutions

It's vital to make the right choices for selection of packages and their accompanying technology, and for business partnerships with package vendors

The effects are not limited to the value-added services. Underlying technology can be affected, business support systems (CRM, provisioning, billing ...) have to handle the changes

You can't get everything right - some solutions won’t work as well as you expected, some vendors will go out of business, some solutions will be overtaken by better ones from competitors

You need an architecture that is not specific to the packages you currently use - an architecture where you can swap old solutions for better ones.
**Agenda**

- Introduction: we need Architecture Management
- The context
- Architecture Management@Swisscom Mobile
  - The approach
  - Some practical experiences
  - Conclusions

**Architecture Management@Swisscom Mobile**

- Define and manage technical Architecture processes within the company
- Develop and maintain technical information (conceptual level)
- Define target Architecture including guidelines, principles and roadmap
- Ensure synchronisation between Architecture roadmap and platform roadmaps
- Control & report on Architecture & Security compliance of solution implementations
- Define Security concepts. Co-ordinate, report & escalate. Responsibility is share with SCM line organisation
- Provide support to the line in understanding Architecture & Security processes and content (End-to-End Architecture & Security)
- Support platform introduction decision according to company priorities
End-to-end architecture management

We define how to integrate systems and capabilities for the greater good of the enterprise, consistently from an end-to-end perspective: From the end-user application to the network element, through business support systems.

We ensure consistency between the architecture layers (e.g.: between Core Network and Enabling) and detect potential conflicts between roadmaps of evolution of different domains.

**Agenda**

- Introduction: we need Architecture Management
- The context
- Architecture Management@Swisscom Mobile
- The approach
- Some practical experiences
- Conclusions
Starting point

The basic ideas:
- Define how to integrate systems and capabilities for the greater good of the enterprise
- Define target architectures and roadmaps
- Synchronise architecture roadmaps with platform roadmaps
- Support projects in identifying platform impact and designing solutions compliant with the architecture
- Review projects in terms of alignment against the defined architecture

Findings:
(We need):
- Descriptions of current (over time changing) and target architectures, which must:
  - Provide reference and content
  - Support control of project architecture compliance
- Architecture Management standardised processes and activities, using this Reference Architecture

Starting point and guide: The Zachman Framework

A framework is a classification scheme that enables focused concentration on selected aspects of subject or object.

It is useful for:
- Simplify for understanding and communication
- Clearly focus on independent variables for analytical purposes
- Maintain a disciplined awareness of contextual relationships that are significant to preserve the integrity of the object

The Zachman Framework is a classification scheme for descriptive representations of complex objects.

It is generic and can be used to classify the descriptive of any object: an enterprise, a product...
The Zachman Framework for Enterprise Architecture : Fundamentals

- **Use the Framework as a classification scheme for descriptive representations of an Enterprise**
- **Different perspectives**
  - Planner : Objectives / scope
  - Owner : Model of business
  - Designer : Model of information system
  - Builder : Technology Model
  - Sub-contractor : detailed representation
- **Different abstractions** (different ways to describe the enterprise)
  - What : Material description - Data
  - How : Functional description - functions and processes
  - Where : Spatial description - Flows
  - Who : Operational description - People / workflow
  - When : Timing description - Dynamics / Events
  - Why : Motivation description - Strategies

---

### The Zachman Framework for Enterprise Architecture - A Framework

<table>
<thead>
<tr>
<th>DataColumn</th>
<th>Data Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Scope</strong></td>
<td>Conceptual</td>
</tr>
<tr>
<td><strong>Vision</strong></td>
<td>Work towards a desired future</td>
</tr>
<tr>
<td><strong>Function</strong></td>
<td>Ensure that the architecture meets the needs of the organization</td>
</tr>
<tr>
<td><strong>People</strong></td>
<td>Focus on the people who will use the architecture</td>
</tr>
<tr>
<td><strong>Time</strong></td>
<td>Consider the time frame in which the architecture will be implemented</td>
</tr>
<tr>
<td><strong>Motivation</strong></td>
<td>Determine the reasons for implementing the architecture</td>
</tr>
</tbody>
</table>

#### The Zachman Framework Model (Logical)

- **System Model**
  - Organizational
  - Information
  - Physical
- **Technology Model**
  - Infrastructure
  - Standards
  - Tools
- **Data Model**
  - Entities
  - Relationships
  - Attributes
- **Process Model**
  - Activities
  - Roles
  - Tools
- **Security Model**
  - Policies
  - Procedures
  - Controls

---

**John A. Zachman, Zachman International**
The Zachman Framework for Enterprise Architecture: Fundamentals

Rules of the framework

- Do not add rows or columns to the framework
- Each column has a simple generic model
- Each cell model specialises its column generic model
- Level of detail is a function of a cell, not a column
- No meta-concept can be classified into more than one cell

Scope

The Zachman framework is not an architecture. It is a classification scheme for organising and managing architecture. We are using it as our reference framework:
- To organise the perspectives and aspects of the architecture we have identified
- To organise what we already have, look for gaps, identify what we need, and prioritise its development
Two Frameworks

Architecture Framework

- Contains reference products that together define the overall architecture
- Each reference product is primitive and contained within one Zachman cell
- Reference products grow incrementally as projects gradually realize the target architecture

Project Framework

- Contains descriptions of project deliverables, which may be composites i.e. may span more than one Framework cell
- Descriptions of deliverables are not prescriptive but there is guidance on what kinds of deliverable would be expected in different types of project
- Framework cells define characteristics and properties that must be present in the deliverables to establish architectural compliance
- Projects produce deliverables that meet the framework specifications
Architect’s roles in projects

- Consultant to Solution Architect – second opinion on solutions
- Support for defining project-specific instance of framework:
  - Kinds of deliverable recommended: may be composites
  - Required properties of deliverables ["You should be able to answer these questions"]: based on reference products, specific to Framework cells
- Support of project team during development:
  - Using reference products as sources of content
  - Using reference products as constraints
- Assessment of architectural compliance

Where have we got to?

- Initial decision: start with CRM and EAI Bus
- One year architecture program management - established control and consistency over initial projects
- Creation of an architecture governance process
- Selection of the Zachman Framework
  - Creation of architecture definition within the Framework
  - Development of guidance
    - for using the architecture definition in projects
    - for ensuring compliance of projects with the architecture
- Applying the guidance on projects
Bridging the gap between project: Architecture Management with the Zachman Framework

Grégory Grin
21-Jan-2004

Agenda

- Introduction: we need Architecture Management
- The context
- Architecture Management@Swisscom Mobile
- The approach
- Some practical experiences
- Conclusions

Step 1: take into account issues due to the specific context (1)

<table>
<thead>
<tr>
<th>Issue</th>
<th>Solution Approach</th>
<th>Framework</th>
</tr>
</thead>
<tbody>
<tr>
<td>There is an “ideal” set of packages that would match closely the capabilities of the conceptual applications - but this set is a moving target</td>
<td>Drive the logical application architecture from business requirements. Maintain mappings to both target and current physical application architecture. Maintain detail of physical architecture in current – keep the target definition “light”.</td>
<td>Logical and physical “How” models</td>
</tr>
<tr>
<td>Delivering required functional capabilities using the facilities of the available packages - may have options to map functions to more than one package.</td>
<td>Make selections and use the Framework to ensure they are applied consistently.</td>
<td>Logical and physical “How” and “Why” models</td>
</tr>
<tr>
<td>Packages need to support business view of data</td>
<td>Enterprise Data Model visible to Business Model owners &amp; mapped to application package databases</td>
<td>Logical and physical “What” models</td>
</tr>
<tr>
<td>Many applications need their own databases and some data has to be replicated</td>
<td>Enterprise Data Model mapped to package data bases. Primary owner application defined for replicated data. Replication defined and managed in physical models.</td>
<td>Logical and physical “What” models, Physical “Where” model</td>
</tr>
</tbody>
</table>
### Step 1: take into account issues due to the specific context (2)

<table>
<thead>
<tr>
<th>Issue</th>
<th>Solution Approach</th>
<th>Framework</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swappable solutions enabled by EAI</td>
<td>Enterprise Message Model to control that the EAI Bus is used consistently across projects.</td>
<td>Logical and physical “Where” models</td>
</tr>
<tr>
<td>It is not possible to phase everything to fit neatly over time. We have to develop interim solutions in some areas.</td>
<td>Interim solutions are swapped in and out in the same way as replacement of parts of the current architecture by parts of the target architecture</td>
<td>Physical “What”, “How” and “Why” models</td>
</tr>
<tr>
<td>Neither projects nor application support teams have the end-to-end view of support for business processes</td>
<td>End-to-end workflow models (with both human and automated actors) are built, and kept consistent with the “How” and “Where” models. These models are owned by the architecture team, and kept in step with the end-to-end view of business process in the business models</td>
<td>Logical and physical “Who” models,</td>
</tr>
<tr>
<td>There are many constraints on the use of packages. Some are inherent in the packages themselves and some resulting from Swisscom Mobile design decisions</td>
<td>Design decisions (and the rationales for them) and package constraints are documented in the “Why” column. They result in constraints and guidance that impact other columns, especially “What” and “How”. It’s vital to make this information readily available to projects in concise form.</td>
<td>Physical “What”, “How” and “Why” models</td>
</tr>
</tbody>
</table>

### Step 2: define the Reference products

<table>
<thead>
<tr>
<th>What</th>
<th>How</th>
<th>Where</th>
<th>Who</th>
<th>When</th>
<th>Why</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Logical Design</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enterprise Data Model (EDM)</td>
<td>Logical Message Model</td>
<td>Logical Workflow Model</td>
<td>Entity Behaviour models</td>
<td>Logical Architecture Rules</td>
<td></td>
</tr>
<tr>
<td>Logical Data Catalog</td>
<td>Logical Data Catalog</td>
<td>Logical Workflow Catalog</td>
<td>Business Event Catalog</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conceptual Application Architecture</td>
<td>Logical Message Catalog</td>
<td>Logical Workflow Actors</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Ownership</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Physical Design</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Application Package DB Specifications</td>
<td>Physical Message Model</td>
<td>Enterprise Workflow Model</td>
<td>System Event Catalog</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EDM Mapping</td>
<td>Physical Message Catalog</td>
<td>System Event Catalog</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Catalog Mapping</td>
<td>Physical Master/Slave Relationships</td>
<td>Integration Manager Specification</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Gregory Grin  
21-Jan-2004
The concept is that each cell in the reference framework should be a view of a single underlying repository.

We have quite a way to go on this – still using Access, Excel, Visio, Word (& some UML tools).
Step 3: Build reference products

Examples of reference products

- Enterprise Data Model
- Logical Data Catalogue
- Enterprise Message Model

Roles of Enterprise Data Model

- The Enterprise Data Model (EDM) is a tool for defining a set of concise, unambiguous, stable, and IT-independent definitions of an organisation’s information resources. It specifies:
  - The significant objects that an organisation needs to hold information about (these are called entities)
  - The way that these objects relate to each other, (these are called relationships)
  - The key facts that the organisation wants to know about these objects (these are often called attributes)
  - The business rules that govern the relationships between these objects

- Enables:
  - Selection of packages that support business views of the data
  - Mapping those views to the packages’ databases

- Supports “primary ownership” of data by conceptual applications - is basis for master/slave data relationships in physical design

- Also used by business people (row 2 owners) for reference
Reference Product: Enterprise Data Model

Informal Overview

The primary purpose of the Logical Data Catalogue is to identify reusable logical groups of data across the whole SCM Architecture on a logical (conceptual) level. In contrast to the physical (technical) level, the logical (conceptual) level:

- is completely implementation independent, i.e.
- clean of design decisions, such as application partitioning, data replication, etc.
- not biased towards any specific technology, such as databases
- is stated in a business readable manner

Data groups are “building blocks”, reusable across business objects, tables, and logical and physical messages

Mapping to messages and physical databases is managed by data group, not by data item
Logical Data Catalogue : Abstract

Primary name: Person

Synonym: individual

Generalisation: entity

Definition: The representation of a natural person (human being).

Name roles: Data definition: Definition: Role constraints

- name: party - data definition: optional
  - id: person - data definition: task
  - name: person - data definition: definition
  - gender: person - data definition: task
  - marital status: person - data definition: task
  - nationality: person - data definition: task
  - date of birth: date - data definition: task
  - language: language - data definition: task
  - occupation: occupation - data definition: task
  - number of children: cardinal - data definition: task
  - date: date of incorporation - data definition: task

Group constraints: (none)

Structure:

Sources: (none)

Informal Overview:

Data groups:

- Person (Orderer)
- Person (Bill Payer)
- Person (Subscriber)
- Party (Orderer)
- Party (Bill Payer)
- Party (Subscriber)
- Party (Subscriber)

- Data group: Personal Information
- Data group: Address Information
- Data group: Financial Information
- Data group: Contract Information
- Data group: Order Information
- Data group: Payment Information
- Data group: Product Information

- Logical Data Catalogue : Abstract

- Bridging the gap between project: Architecture Management with the Zachman Framework

- Gregory Grin
  21-Jan-2004
Reference Product example: Logical Message

The primary purpose of the Enterprise Message Model is to identify reusable message types across the whole SCM Architecture on a logical (conceptual) level.

<table>
<thead>
<tr>
<th>Item roles</th>
<th>Data definition</th>
<th>Definition</th>
<th>Role constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>person id</td>
<td>Person Identifier</td>
<td>Identifier to uniquely identify the person who’s address has changed</td>
<td>• optional</td>
</tr>
<tr>
<td>organisation id</td>
<td>Organisation Identifier</td>
<td>Identifier to uniquely identify the organisation who’s address has changed</td>
<td>• optional</td>
</tr>
<tr>
<td>change</td>
<td>Change Type</td>
<td>(as defined in data definition)</td>
<td>• mandatory</td>
</tr>
<tr>
<td>effective date</td>
<td>Date</td>
<td>The date on which the requested change becomes valid.</td>
<td>• mandatory</td>
</tr>
<tr>
<td>physical address</td>
<td>Physical Address</td>
<td>(as defined in data definition)</td>
<td>• mandatory</td>
</tr>
</tbody>
</table>

Group constraints: • defined(person id) xor defined(organisation id)

Producers: • CRM
Consumers: • OM
• Billing
• Collection
• Provisioning

Agenda

B Introduction: we need Architecture Management
B The context
B Architecture Management@Swisscom Mobile
B The approach
B Some practical experiences
B Conclusions
Conclusions

- The Zachman Framework provides a holistic view on the enterprise and helps to identify gaps
- The Zachman Framework is even then helpful, when you plan to use commercial packages instead of developing the IT infrastructure by yourselves
- Adapting the Zachman Framework to company-specific needs allows the definition of artefacts to be delivered by projects required for architecture governance
- Reference products provide a “quick-start” solutions for required artefacts
- Reference products are highly reusable and reusability should be a primary goal
- Project products are means to check architecture compliance
- Projects must be treated as “customers” from the architectural governance perspective
- A common language (such as UML) is very helpful
- Coaching/mentoring on Zachman and architecture is helpful and much more efficient in changing the culture than direct support.

Thank you for your attention

Grégory Grin
Head of Architecture Governance
gregory.grin@swisscom.com

John Hall
Consultant
john.hall@knowgravity.com