Engineering Virtual Domain-Specific Service Platforms

Specific Targeted Research Project: FP7-ICT-2009-5 / 257483

Architecture for Role-Based Governance of Virtual Service Platforms

Abstract

For the design, implementation and operation of Virtual Service Platforms good governance is indispensable. This is especially important as two technological approaches come together in INDENICA: Product Line Engineering and Service Orientation. From an ecosystem of governances we select a governance framework, evaluate its suitability using a set of requirements. For the specific aspects of INDENICA platforms roles and responsibilities, governed processes and guidelines are elaborated in detail. For monitoring governance a number of Key Performance Indicators are derived from the platform policies and can thus be used for creating input to the monitoring and adaptation engines.

Document ID:	INDENICA – D3.2
Deliverable Number:	D3.2
Work Package:	3
Type:	Deliverable
Dissemination Level:	PU
Status:	final
Version:	1.0
Date:	2012-02-18
Author(s):	SIE, PDM, TEL, TUV

Project Start Date: October 1st 2010, Duration: 36 months

Version History

1.0	18 Feb 2012	First submitted version

Document Properties

The spell checking language for this document is set to UK English.

Abbreviations

BPM	Business Process Management
COBIT	Control Objectives for Information and Related Technology
COTS	Commercial of the Shelf
EA	Enterprise Architecture
IT	Information Technology
LOB	Line of business
PLE	Product Line Engineering
PMP/QMP	Project Management Plan / Quality Management Plan
QMS	Quality Management System
SGF	SOA Governance Framework
SGVM	SOA Governance Vitality Method
SOA	Service Oriented Architecture
TOGAF	The Open Group Architecture Framework
VSP	Virtual (Domain-Specific) Service Platform

Table of Contents

1 Introduction

1.1 Motivation

A service platform consists of infrastructure assets, like communication middleware or databases, and platform services that together constitute the interface and programming model for application service development. Building domain-specific service platforms is necessary to fulfil the specific requirements of the various domains that are sometimes incompatible with each other and ease service and application development within the domain. These domain-specific service platforms together form a family of platforms, where members share assets or where differences are explicitly designed into the members. In this way, the INDENICA approach tailors the platforms towards the application domains and provides methods and tools for designing and implementing a Virtual Domain-Specific Service Platform (VSP).

This approach not only requires novel technological approaches but also on the implementation and application environment from an organisational and human behaviour point of view. Design, development, integration, test, deployment, are all tasks performed or monitored by people. In order to achieve corporate goals and economic performance, processes, rules, guidelines and respective controls need to be in place. All these aspects are summarized under the term “governance”.

When introducing a platform in traditional product development this imposes in the beginning a higher effort for design and development that in just developing independent products. In order to take benefit of such platform approaches, products based on the platform have to follow the reference architecture and design and implementation guidelines. Thus the architecture governance is the backbone of the system.

Similarly to a Product Line Engineering approach the challenges on a coordinated and successful implementation of VSPs are higher than just introducing a single service platform. And – in addition to this – the paradigm of service orientation imposes even more need for having good governance.

In the “Report on State of the Art in Service Platform Design, Adaptation, Deployment and Monitoring” [INDENICA D1.1] we gave an overview on the great number of descriptions of SOA Governance in literature and industrial practice. But we already saw deficiencies – especially in architecture governance – that we want to eliminate with the definition of an INDENICA Governance.

In addressing these issues, this document shall be a guide for all who are involved in the decision to introduce platforms and are responsible for the implementation. This holds especially for executives like Corporate Information Officers or IT Strategists, for architects like Chief Architects, Platform Architects and for Process Engineers.

1.2 Objectives

In this document we will outline an approach for the governance of virtual service platforms.

A as first step, in chapter 2, we will analyse various definitions of governance that are present in literature and standards: IT Governance, SOA Governance, EA Governance, Architecture Governance and some more are definitions which are often concurrent or overlapping. An ecosystem of governance will be sketched, that embraces many kinds of governance and tries to bring them into a structured relationship.

I a further step (chapters 3) we will look at the requirements on a Governance Framework and analyse standards and literature, if such frameworks are already in place.

The main part of this document will then be the elaboration of a Governance Framework for Virtual Service Platforms (chapters 4 and 5). It will contain role model, where we describe in detail the roles introduced for designing a VSP and managing its variability. Then we will look at core processes like portfolio and lifecycle management of platforms, and the principles for architecture governance.

As any set of rules or processes, also governance must be adapted regularly to changing technological, organisational and business environment. A continuous improvement is necessary in order to ensure impact and acceptance. For this sake we define a set of Key Performance Indicators (KPI) based on policies that were sketched in the INDENICA Case Studies, and which help monitoring the behaviour and performance of the platforms.

Overall, the work on INDENICA governance shall help organisations taking the VSP approach to create awareness on the need for suitable processes, for skilled people and for an improvement cycle based on monitoring results. Beyond the awareness, the described roles, processes and KPIs shall give a guideline how such governance can be implemented and how evidence of impact and benefit can be given.

1.3 Relationship with Other INDENICA Work

In previous INDENICA deliverables some aspects of governance were touched without going into detail. In “View-based Design Time and Runtime Architecture for Tailoring VSPs” [INDENICA D3.1] roles and a rough development process were introduced that are relevant for setting VSPs, and in “Description of Feasible Case Studies” [INDENICA D5.1]some governance requirements and a number of possible policies were described.

These roles and policies are input for a deeper analysis of governance aspects. They will be described in more detail, allocated to their position and reference point in a Governance framework.

The refined role model will then be input for the further work on the INDENICA tool suite guidelines. The Key Performance Indicators will be input for the monitoring engine in terms of rules and to be formally described. Last but no least the governed processes, especially the Change Management Process can be applied while validation the IDENICA methodology and tools along the Case Studies.

The flow of outputs / inputs is depicted in Figure 1.

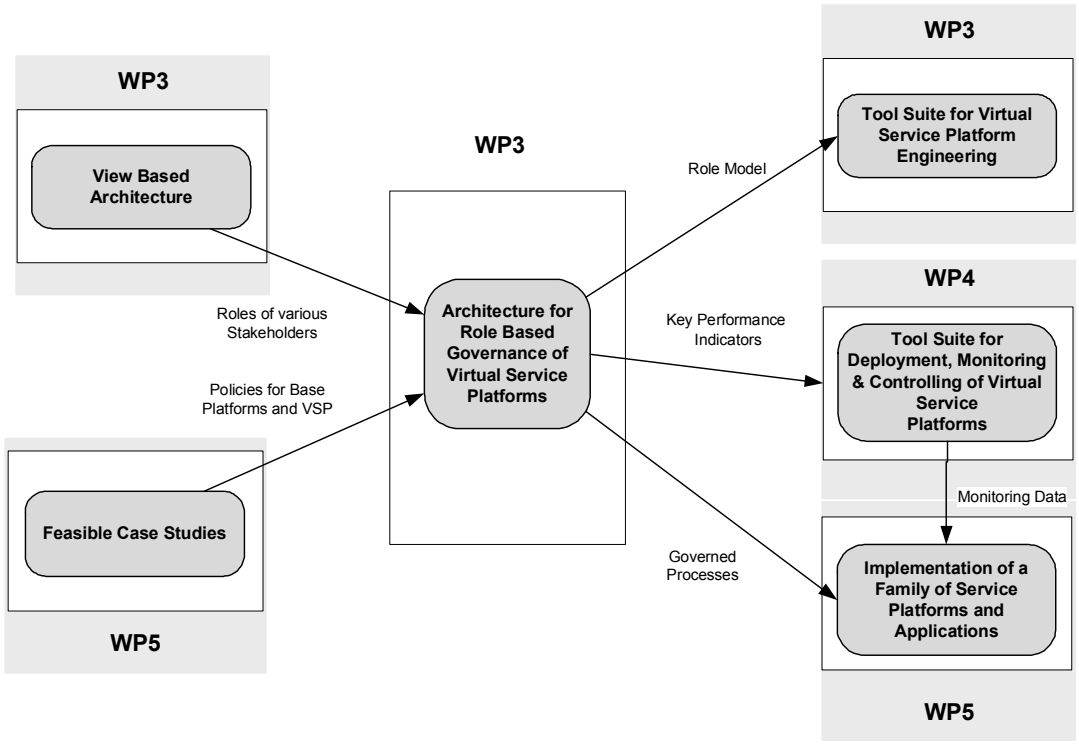

Figure 1 Relationship with other INDENICA work

2 Eco-System of Governances

In an enterprise different kinds of governances coexist. In the context of INDENICA the following types of governance are relevant:

- Corporate Governance
- Business Process Management Governance (BPM Governance)
- Information Technology Governance (IT Governance)
- Enterprise Architecture Governance (EA Governance)
- SOA Governance
- Architecture Governance

In order to get a good understanding of governance in general and of the different types of governance, the purpose of this chapter is to define the different types of governance and to show the relations between these different governance types. Further it will introduce the SOA Governance Framework of the Open Group, which will be the base of the INDENICA Service Governance Framework.

2.1 Overview and Definitions

A comprehensive definition of governance was elaborated in The Open Group's draft technical standard Service Oriented Architecture Ontology [Ontology 2009]: "*The term 'governance' is originally from political theory, where it refers to a system by which a political unit is controlled, and to the exercise of that control. The term is now also used in relation to enterprises, where it applies to all aspects of enterprise operation, including architecture development and implementation. Good governance is widely recognized as being crucial for successful deployment of SOA".*

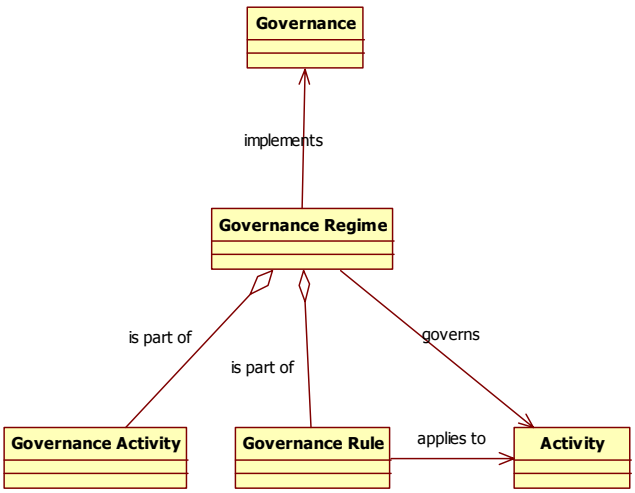

Figure 2: Governance and Governance Regime

In the context of governance definitions also the terms "Governance Regime", "Governance Rule" and "Governance Activity" are frequently used. Figure 2, which is

derived from a figure in [Ontology 2009] p. 78, shows the relations between these aspects.

A governance regime is the implementation of governance in a concrete environment, e.g. an enterprise or a subdivision. It consists of governance rules and governance activities and controls all other activities of the organisation.

2.1.1 Types of Governance

Within an enterprise there are several types of governance in place that are not independent from each other. The relation of the different types of governance and their relationships are shown in Figure 3.

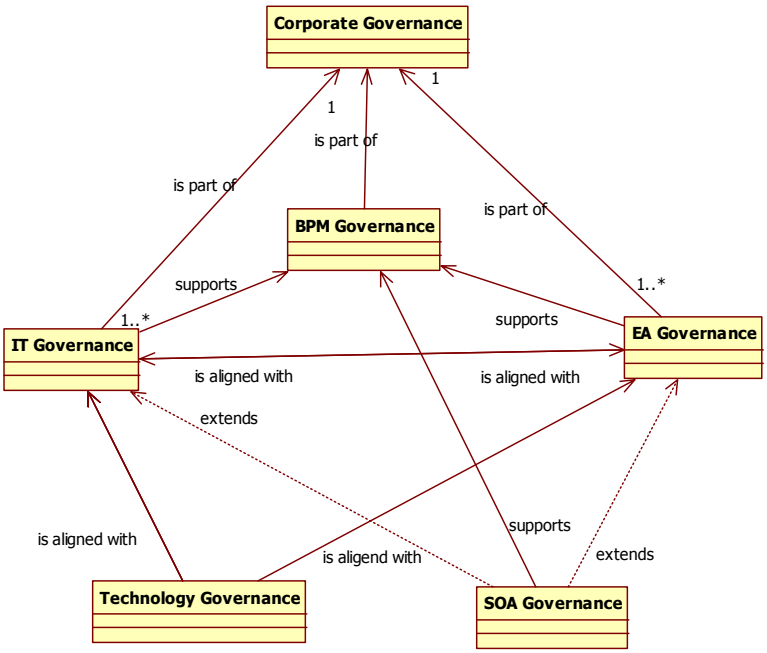

Figure 3: Relations between the different types of governances

All these types of governances should follow similar approaches and are implemented as part of the overall Corporate Governance model, so all these governance types support Corporate Governance.

Corporate Governance

Wikipedia defines corporate governance as: *„Corporate governance consists of the set of processes, customs, policies, laws and institutions affecting the way people direct administer or control a corporation."* [Wikipedia 1]

In this scenario Corporate Governance focuses on the rights, roles, and equitable treatment of shareholders, defines disclosure and transparency, and ensures:

- Strategic guidance of the organization,
- Monitoring of management by the board,
- Board accountability for the company and to the shareholders.

The Organization for Economic Co-operation and Development states: *"Corporate governance involves a set of relationships between a company's management, its board, its shareholders and other stakeholders. Corporate governance also provides the structure through which the objectives of the company are set, and the means of attaining those objectives and monitoring performance are determined. Good corporate governance should provide proper incentives for the board and management to pursue objectives that are in the interests of the company and its shareholders and should facilitate effective monitoring. The presence of an effective corporate governance system, within an individual company and across an economy as a whole, helps to provide a degree of confidence that is necessary for the proper functioning of a market economy."* [OECD 2004] p. 11

In the context of INDENICA the term "Corporate Governance" is used instead of "Enterprise Governance".

As BPM Governance extends Corporate Governance and SOA Governance extends IT Governance and EA governance, they must be aligned with each other.

Sometimes the Corporate Governance itself is affected by changes in BPM / IT / SOA Governance. For instance, the introduction of a CIO, the reorganisation of teams to better address architecture and platform issues (platform teams and application teams) are decisions and changes that have to be treated under an overall enterprise aspect. In this way Corporate Governance is not only supported by BPM, IT and SOA Governance, but also dependent on them.

BPM Governance

BPM Governance is a set of policies, roles, responsibilities and processes that set the way how an organization's business processes are run. Key elements of good BPM governance include transparency, responsibility and accountability, and commitment to the organization's business goals.

In this way BPM Governance is a main constituent of Corporate Governance beneath others like Values, Compliance, Communication, or Environment, Health and Security.

In today's enterprises business processes are to a large extent represented in their information technology system containing infrastructure, data and applications. Here two more definitions of governance come up: IT and EA Governance.

IT Governance

IT Governance is a subset discipline of Corporate Governance focused on information technology (IT) systems and their performance and risk management. The widely used and referenced IT Governance Framework COBIT [COBIT 4.1] defines it as: *IT Governance includes the decision rights, accountability framework and processes to encourage desirable behaviour in the use of IT.*

Today there are virtually no enterprises in place that do not depend on IT infrastructure[1]. So it is not of surprise that IT Governance was the first governance to be described, structured and standardized: COBIT was first released in 1993.

In the context of the ecosystem of governances, IT Governance is also a part of corporate governance. But I shall not be in concurrence with BPM Governance but support it in order to facilitate the core processes of the business like Customer Relationship Management, Supply Chain Management, or Product Lifecycle Management.

EA Governance

The Open Group Architecture Framework ([TOGAF 2011], see chapter 2.3) gives the following definition: "*EA Governance is the practice and orientation by which enterprise architectures and other architectures are managed and controlled at an enterprise-wide level*."

The architectures that EA Governance addresses are related to the business, the data, the applications and technology (cf. Figure 4). Business Architecture and Application Architecture are closely related to BPM and in this way EA Governance shall give a strong support to BPM Governance. At the same time Data Architecture and Technology Architecture are also part of the IT landscape and here the need for strong alignment between IT- and EA-Governance arises.

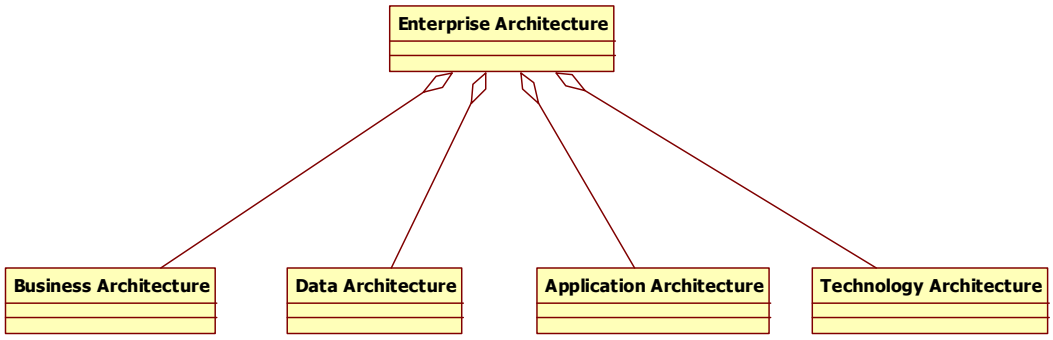

Figure 4: Elements of Enterprise Architecture (according to TOGAF)

The most influencing parts of this EA model on the introduction of VSPs are data, application, and technology architecture.

Based on TOGAF Siemens has developed an Enterprise Architecture Management Methodology that offers an overview about the overall platform architecture, ranging from the actual business functions over integration and infrastructure layer to supporting guidelines and processes. This method was described in INDENICA Deliverable D3.1, chapter 2.2 and 5.4.

SOA Governance

There are many different definitions of SOA Governance in literature. The Software Engineering Institute gives the following: *"SOA Governance should be viewed as the*

[1] There might be some micro-enterprises or self-owned businesses as an exception, but those are not part of our considerations.

application of Corporate Governance, IT Governance and EA Governance to Service Oriented Architecture. In effect, SOA Governance extends IT and EA Governance ensuring that the benefits that SOA extols are met. This requires governing not only the execution aspects of SOA but also the strategic planning activities." [SEI 2009]

Wikipedia gives a very generic definition: "*SOA Governance is a concept used for activities related to exercising control over services in an SOA. SOA governance can be seen as a subset of IT governance, which itself is a subset of corporate governance"* [Wikipedia 2]

Many of these definitions overlap or are contradictory in some aspects. The most comprehensive work on SOA Governance is the draft SOA Governance Framework from The Open group, which we will introduce in chapter 2.3.

As SOA is a technology that supports the definitions and implementation of business processes, but also applications, data and technology, it is an extension to IT Governance and EA Governance.

Technology Governance

Technology governance controls how an organization utilizes technology in the research, development, and production of its goods and services. Although it may include IT governance activities, it often has broader scope [TOGAF 2011].

In the ecosystem of governances it needs a strong alignment with IT Governance and with EA Governance, and thus be a driver for principal decisions, e.g. for introducing a SOA Approach.

2.1.2 Architecture Governance

All definitions in chapter 2.1.1 have a strong enterprise and organisation direction and they also address aspects of system and platform architecture.

But we see the need for defining also an Architecture Governance for two main reasons:

1) Platforms are on one side introduced as infrastructure for EA, IT and SOA and in this way support the definition and implementation of and enterprise's business processes. But platforms are often also part of a Product Line Engineering (PLE) approach for product and solution development and thus the backbone for the lifecycle management and value chain. This aspect is not really represented in the above listed definitions.
2) Even platforms that are part of a company's products and used by thousands of users and application developers, are not sufficiently covered. An example is today's cloud computing platforms from Amazon, like EC2 or S3. Amazon not only provides services, quality of service and a pricing model, but also governance in order to ensure appropriate and conflict-resistant usage of its platforms[2].

[2] Note that there is no single governance document or website, but a number of guidelines (see http://aws.amazon.com/documentation/)

Architecture Governance is a systematic approach for managing architectures and controlling all modifications in order to ensure quality and sustainability. This holds for all modifications, those for developing the system and those for evolving it.

The Open Group defines in its Architecture Capability Framework (Part VII of TOGAF) Architecture Governance as *"the practice and orientation by which enterprise architectures and other architectures are managed and controlled at an enterprise-wide level"* [TOGAF 2011]. Thus it does not only address the high-level Enterprise Architecture, but also the architecture of systems, sub-systems and platforms.

Thus Architecture Governance includes the following:

- Implementing a system of controls over the creation and monitoring of all architectural components and activities, to ensure the effective introduction, implementation, and evolution of architectures within the organization
- Implementing a system to ensure compliance with internal and external standards and regulatory obligations
- Establishing processes that support effective management of the above processes within agreed parameters
- Developing practices that ensure accountability to a clearly identified stakeholder community, both inside and outside the organization.

2.2 Requirements on Governance Frameworks

As mentioned above there is plethora of work on different kinds of governance, but all contradictions and overlapping definitions give the impression that – besides corporate governance – governance at all is a fast moving target, an unchartered area. One example for this is the paper on a BPM Governance Framework from Vitaly Khusidman which lists nine requirements and expresses the expectation that this list will be updated and enhanced [Khusidman 2010]:

Requirement 1 - Single point of reference

Requirement 2 - Coexistence with other Governances

Requirement 3 – Guiding Principles

Requirement 4 – Governed and governing aspects

Requirement 5 – Guidelines for governed aspects

Requirement 6 – Guidelines for governing aspects

Requirement 7 – Organization Alignment

Requirement 8 – Standards

Requirement 9 – Continuous Improvement Method

Khusidman further analyses different approaches for a BPM Governance Framework, and evaluates these frameworks according to this list of requirements. Here he comes to the point where he recommends The Open Group SOA Governance Framework [SGF 2009] to be the best approach.

This might seem contradictory, as this framework understands itself as focused on SOA Governance, and Khusidman refers to BPM Governance. But his requirements can be seen as generic for any kind of governance.

In requirement 4 Khusidman makes a clear distinction between governing and governed processes. TOGAF suggests a number of governing aspects, where the Open Group SGF covers only three: Communication, Compliance and Dispensation. From the four missing ones[3] "Monitoring and Reporting" is of great importance for INDENICA VSPs; we will elaborate on this in chapter Monitoring INDENICA Governance.

Requirement 9 refers to an improvement method which in The Open Group SGF is called "SOA Governance Vitality Method". Khusidman is adopting the approach and thus acknowledging that The Open Group SGF is fulfilling requirement 9. For platform development and maintenance also a CMMI® -based approach could be appropriate

In chapter 5.3 we will detail a part of the improvement cycle based on the INDENICA Key Performance Indicators.

Other requirements

In [SEI 2009] the Software Engineering Institute (SEI) proposes a structure of a SOA Governance Framework using an entity relationship diagram.

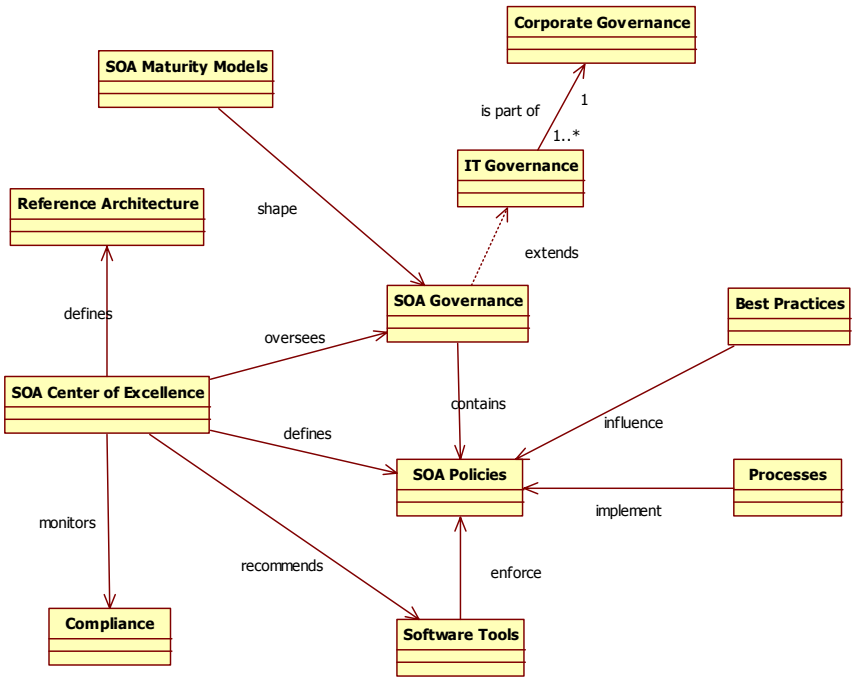

Figure 5: SOA Governance Framework Elements (derived from [SEI 2009])

Requirement 10 – Reference Architecture

[3] The other missing ones are: Policy Management and Take-On, Business Control, and Environment Management

The architecture of a SOA Governance Framework presented in Figure 5 introduces four main aspects that were described in literature before, but have never been put in such a defined relationship. Especially the responsibility of the SOA Centre of Excellence for a Reference Architecture is of importance for INDENICA; here we suggest adding this new requirement specific for platform governance.

Requirement 11 – Risk Management

The INDENICA project brings together two paradigms that have been treated up to mow independently: Product Lien engineering and Service Oriented Architecture. Inherently to such an approach there are specific risks that have to be analysed and treated. Risk Management in general has two aspects: product related risks and business risks. Product related risks address safety and security aspects and in many domains there are respective standards in place that will then be part of the Architecture Governance and its guiding principles. In chapter 5.2.4 we will address this issue and refer to an example of a guideline for agile development in regulated environments.

Risk mitigation is one part of risk management. For platform approaches and especially for VSPs, prototyping, simulation etc. could be identified as methods for that. A deeper analysis of these methods for VSPs and their impact on risks and dependability is not part of the INDENICA work and leaves room for further research.

2.3 The Open Group SOA Governance Framework

Khusidman sees the applicability of The Open Group SGF on a high level but stipulates that for a specific domain, specific artefacts and templates shall be defined [Khusidman 2010].

For INDENICA this would mean that we choose this existing SOA Governance Framework and enhance it with specific aspects related to Virtual Domain Specific Service Platforms.

In the following we give an overview on structure and content on The Open Group SGF.

2.3.1 Overall Structure

The structure of the SOA Governance Framework of The Open Group [SGF 2009] is shown in Figure 6. It consists of

- a SOA Governance Reference Model
- a SOA Governance Vitality Method.

It will be the base of our further considerations.

Using the SOA Governance Vitality Method an organization customizes the SOA Governance Reference Model for its specific requirements resulting in a SOA Governance Regime.

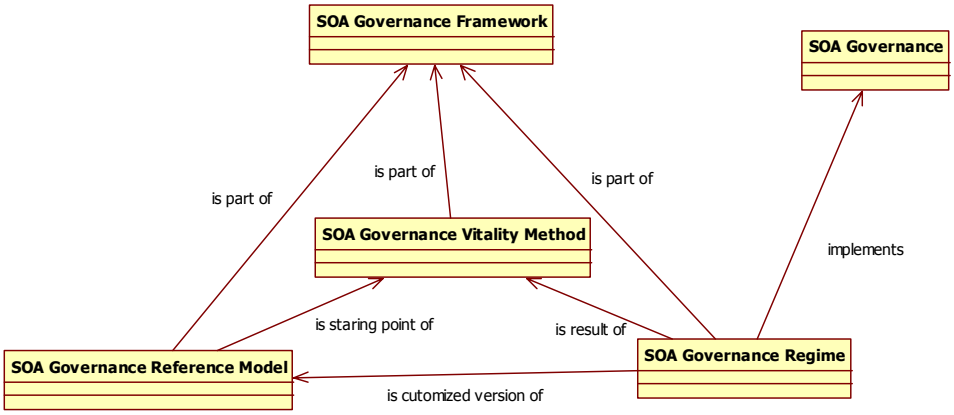

Figure 6: Elements of the Open Group SOA Governance Framework

The SOA Governance Reference Model has the following structure (cf. Figure 7):

- SOA Governance Guiding Principles
- SOA Governing Processes, that are:
  - Compliance
  - Dispensation
  - Communication

  In addition to these Khusidman suggests the following processes:
  - Policy management and Take-On
  - Monitoring and Reporting (which could also be part of compliance)
  - Business Control (which could also be part of compliance)
  - Environment Management (ensuring that the environment of the governance framework is effective and efficient)
- Governed SOA Processes
  - Service Portfolio Management
  - Service Lifecycle
  - Solution Portfolio Management
  - SOA Solution Lifecycle
- SOA Governance Artefacts
- SOA Governance Roles and Responsibilities (Khusidman has some aspects concerning organization in his article)
- SOA Governance Technology

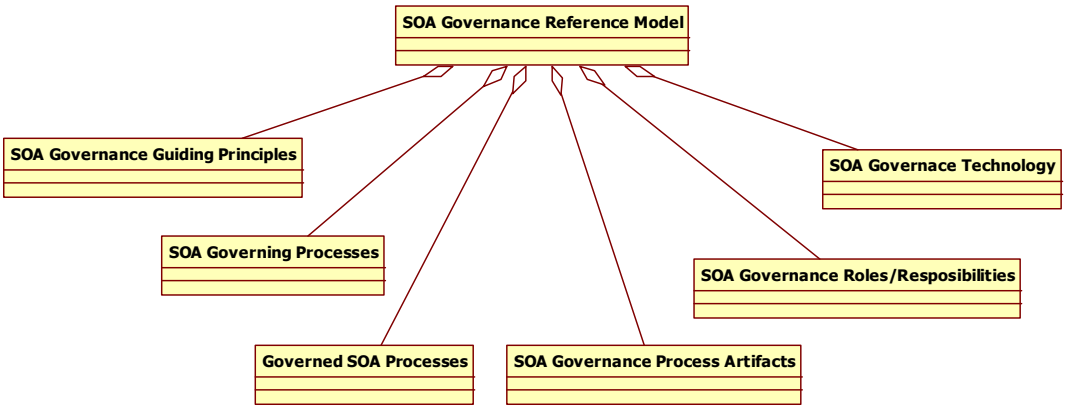

Figure 7: Elements of the Open Group SOA Governance Reference Model

The Open Group SGF defines a SOA Governance Vitality Method for continuous improvement thus covering above mentioned Requirement 9. The SGVM improvement cycle consists four phases: Plan – Define – Implement – Monitor. This approach from a structural point of view is also not specific to SOA and governance, but similar to other improvement cycle definitions like the Deming Cycle PDCA [Deming 1986]. In the INDNEICA context we will go one step deeper and elaborate in chapter 5.3 and 5.4 the relevant Key Performance Indicators and the improvement cycle for VSP and the role of platform monitoring and platform adaptation.

2.3.2 Guiding Principles

Guiding Principles are abstract rules that represent a set of values and best practices. Usually they are not yet measureable. Which guiding principles are selected and how strictly they are applied depends on the governance maturity of an organization. The Open Group SOA Governance Reference Model defines a set of guiding principles, e.g.

- A SOA Reference Architecture is required (addressed by the INDENICA View-Based Architecture)
- Service reuse (addressed by the VSP)
- Service monitoring (addressed by the INDENICA governance model)

2.3.3 Processes

The SOA Governance Reference Model differentiates between governing and governed processes.

The *Governing Processes* include compliance, dispensation and communication processes. The objective of the compliance processes is to ensure adherence to policies, guidelines, and standards defined by the INDENICA governance model. There are two categories of policies. Policies used to govern services prior to deployment are called *design-time* policies. Policies for describing the correct behaviour of service operation are called *run-time* policies.

The *Governed Processes* include planning, design and operational aspects of Solution Portfolio Management, Service Portfolio Management and Lifecycle Management.

2.3.4 Roles and Responsibilities

The SOA Governance Reference Model of the Open Group defines a set of boards (e.g. SOA Steering Board, EA Governance Board or SOA Centre of Excellence) and teams (Solution Development Team and Service Development Team) based on several key roles within the organization.

The following table lists the teams and roles defined in the SOA Governance Framework [SGF 2009], p. 29. In order to ease further work, we list here the complete table, but applied a different ordering of items:

Team *(Function)*	Participating Roles	Responsibilities of the Team
Business Domain Representative *(Scope and Delivery Management)*	• Program Manager • Business Architect • Process Engineer	• Responsible for the solution from a business perspective by justifying the solution and services existence and continuous operation to the stakeholders • Determine business service functionality • Communicate business requirements and identify business services for each domain • Share information regarding specific business requirements and identify the cross-organisational SOA business services • Work on prioritizing program requirements and services • Develop service proposals to go through funding process
IT-Executive Steering Board *(Sponsorship of all IT-Solutions and Services)*	• CIO • CTO of Chief IT Strategist • Chief Architect • Business Domain Owners	• Ultimate decision makers for decisions regarding SOA solution, service and IT related matters • Approve SOA strategy direction • Approve governance Principle
SOA Steering Board *(Sponsorship and Leadership of SOA Program)*	• SOA Chief Architect • SOA Director • SOA Business Sponsor	• Define future SOA strategic direction and roadmap • Monitor SOA strategic direction • Ensure that SOA principles and practices will make an appropriate and necessary contribution to the overall enterprise business strategy • Support the desired outcomes and objectives by providing funding and resources for the SOA and SOA

		Governance • Defines the SOA Governance principles
EA Governance Board *(Solution and Service Lifecycles*)	• Chief Enterprise Architect • Enterprise Architects • Chief SOA Architect	• Define and develop the service portfolio • Define and develop the SOA solution portfolio (segment/ domain architecture)
SOA Centre of Excellence *(Definition and Development*)	• Chief SOA Solution Architect • Organizational Change Consultant • Test Strategist • SDLC Responsible • Project Management Process Responsible • Operational Process management Responsible • Tools Strategist	• Collaborate to develop SOA governance roadmap, transition plans and governance principles (SGVM) • Definition and development of SOA governing processes and best practices • Definition and development of governed SOA processes and best practice • Define where the compliance checkpoints should be inserted into governed SOA processes • Definition and monitoring of SOA metrics across the LOBs (SOA Governance KPIs) • Architectural definition and integration support across LOBs (consult) • Initiate SOA and SOA Governance organisational changes • Develop governed SOA transformation plans • Identify SOA training and mentoring plans • Define and validate changes to the project management process • Select and implement the SOA Governance tool strategy
SOA Governance Board *(Informing and Monitoring)*	• SOA Chief Architect • Business Architects	• Ensure compliance with standards and guidelines • Dispensation • Communication
Solution Development Team *(Execution and Delivery)*	• Project Manager • Business Analysts • Solution Architects • Integration Specialist • Operations Architect • Developers • Testers • Security Architects	• Manage the solutions within a specific domain • Design, development, testing, deployment, execution and delivery of the SOA solution within the domain • Maintain consumer side interfaces to services • Follow standards and guidelines • Understand an abide by the governing process

Service Development Team *(Execution and Delivery)*	• Project Manager • Business Analysts • Service Architects • Integration Specialist • Operations Architect • Developers • Testers • Security Architects	• Design, development, testing, deployment, execution and delivery of the services • Maintain interfaces to its services • Follow standards and guidelines • Understand an abide by the governing process
IT Operations *(Execution and Delivery*	• Database Administrator • Network Infrastructure Architect • System Administrator • Operations	• Database administration services support • Network infrastructure services support • System administration support • Support for central IT functions • Follow standards and guidelines • Understand an abide by the governing process

Table 1: SOA Governance Roles and Responsibilities, derived from [SGF 2009] page 29

In the following chapters we will make proposals to enhance this set of roles and boards and elaborate changed items. The EA Governance Board will be renamed to EA Board and shall also comprise business roles, not only architects.

The introduction of Virtual Service platforms will bring the need to define additional roles and allocate them to these boards.

3 Governance in the Context of INDENICA

The aim of INDENICA is to provide means for building virtual platforms spanning different application areas as well as different levels of the automation pyramid (see [INDENICA D1.1] p. 35). In this context it's essential to consider not only the IT and SOA aspects but also the business processes which should be supported by these virtual platforms. Examples for such business processes are described in the INDENCIA Case Studies [INDENICA D5.1]: Warehouse Management, Yard Management, Remote Maintenance and their integration.

Having a multitude of roles, teams, projects, and tools concurrently working on an enterprise BPM initiative is a major motivation for establishing an integrated governance view also containing BPM Governance within the INDENICA Service Governance Framework.

For developing consistent (reference) architecture and ensuring the appropriate usage, evolution, adaptation and modification of a Virtual Service Platform, architecture governance in INDENICA is indispensable. Without such governance the VSP and its base platforms would quickly become compromised by wrong deployment of services, wrong usage of services, project failure, over-complex applications, buggy implementations, and design erosion; and eventually to dissatisfied customers and users (and dissatisfied architects and developers).

In this chapter we will derive the need for processes, roles and responsibilities and policies as the main influencing drivers while implementing a VSP.

3.1 Processes

3.1.1 Governing Processes

In the context of INDENICA there are different levels of platforms to be considered, the base platforms, the domain specific platforms and the VSP. First, policies on the VSP level have to be defined. In a second step these VSP policies are refined and mapped to the domain and base platforms. For these platforms sets of policies are already available as a result of the governance activities on this level. The refined policies have to be consolidated with these policies resulting in updated sets of policies on this level.

Governing principles on the VSP level that influence the existing platforms are e.g.

- The request for compliance with business rules and regulations like Sarbanes-Oxley Act or Basel III. From these a derived and measurable policy for e.g. the warehouse management system is to have at any time an overview on the bound capital in the warehouse.
- A quality management system must be in place that ensures adherence to policies and guidelines. It shall also comprise compliance to relent instances of governance.

- Base platforms and technology vendors must be chosen in a technology selection process.
- All services are subject to the Service Portfolio Management process and the request for additional or new services has to be forwarded to the EA Governance Board for approval and decision to implement or purchase a new service.
- All available services have to be published in a centrally accessible service repository.
- Services exposed by the existing platforms should be used whenever appropriate. For approval of exceptions the SOA Governance Board should be asked.
- All services should provide variability in order to be tailored to the application developers' needs.

Examples of additional policies on both levels will be elaborated in chapter 3.3.

3.1.2 Governed Processes

[SGF 2009] lists a number of governed processes out of which the following are relevant in the context of Virtual Service Platforms:

The Service Portfolio Management has to address the variability aspects in the scoping phase and has to decide which services should be available on the VSP. The Solution or Application Portfolio Management ensures that the organization has a set of applications appropriate to satisfy its needs.

The Service Portfolio Management and Lifecycle Management on the VSP level should be consolidated with the corresponding processes of the existing platforms.

3.2 Roles and Responsibilities

In the context of the INDENICA case studies the main focus is on roles involved in the integration of the existing platforms. These are key roles of the Solution and Service Development Teams. A set of roles is already addressed by the INDENICA deliverable on View-Based Architecture [INDENICA D3.1]:

- Platform Provider
  is a technology expert and describes the current variability and the variability binding process of the existing platform he owns.
- Platform Variant Creator
  is responsible for binding unresolved variability in base platform(s) and for creating an executable platform variant
- Platform Architect
  is responsible for VSP requirements, variability within VSP, baseline architecture and adaptation behaviour of VSP
- Platform Integrator
  generates the VSP instance.

- Application Developer
  develops applications based on the VSP instance
- Platform Administrator
  Monitors the current state of the existing platforms and is responsible for making adaptation decisions for the VSP instance

As there are two levels of platforms within INDENICA, the boards and teams of the higher level should include roles of the existing platform level and vice versa. Those central aspects will be elaborated in chapter 5.1.

3.3 Policies

Policies are concrete and measurable rules that are often mapped to non-functional requirements. In INDENICA a number of policies can be derived from the Case Studies as described in [INDENICA D5.1]. The case studies describe three base platforms covering a Warehouse for storing and retrieving goods, the Yard covering registration of vehicles, loading bay and parking area, and the remote maintenance system for supporting warehouse and yard staff in repairing and maintain the systems.

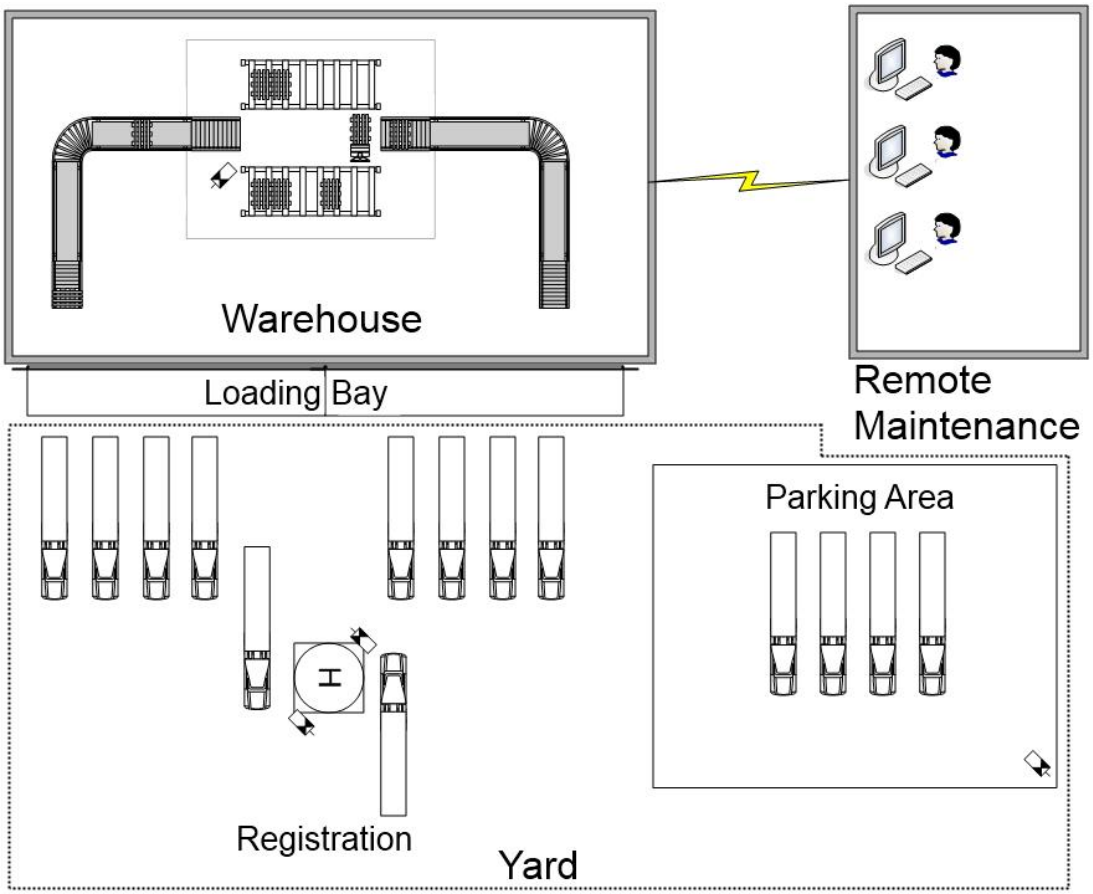

Figure 8 INDENICA Case Study Overview (taken from [INDENICA D5.1])

In the following paragraphs we will describe a number of policies related to these case study systems and their integration. All these policies and related requirements are meant for illustration and shall not be transferred into existing systems without a clear definition process (using e.g. scenario based methods like utility trees).

Warehouse Subsystem

There are two sources for policies in the warehouse: the governing process of the warehouse management subsystem and the set of refined policies derived from the VSP policies (for examples see 3.3). Based on this a consolidated set of policies is defined during the integration process.

Examples for additional Warehouse Management Subsystem policies regarding the usage of the sample services are e.g.:

- The Warehouse Management Service utilisation must be registered at the warehouse platform owner.
- The usage of the transport control service is restricted to the warehouse internal applications and must be protected by appropriate authentication.
- The rejection rate of storage requests should be less than 1%. This leads to the following monitoring and dispensation activities:
    - Keep the capacity utilisation of the warehouse below 95 %. If utilisation exceeds 95% notify warehouse management, who shall take remediate action.
    - If the storage request for a new item fails, retry after 5 minutes. If it fails again, notify the warehouse operator and stop all incoming items until the operator releases operation.
- Retrieval jobs have to be processed en-bloc.
- The rate of non-retrieval of requested items shall not exceed 0.1%.

Remote Maintenance Subsystem

To ensure adherence to run-time policies on the VSP and the existing platforms level, monitoring activities are required. These activities are in the duty of the Remote Maintenance Subsystem.

Monitoring will be based on the supporting monitoring and adaptation framework. A Complex Event Processing engine will be used to detect and aggregate event and later to produce benchmark reports, which will be stored in the repository. Based on these reports, data mining techniques will be used to gather intelligence that will also take part in near-real time analysis. The whole monitoring of the platforms can be seen as a closed loop that self-adopts to current state in time.

Standard message formats will be defined to provide a scalable environment, which can be easily extended with additional platforms.

From the management/supervision point of view, JMX technology will be used to pass adaptation directives and create policy-based management solution.

To govern the development and implementation of the Remote Maintenance Subsystem a set of policies is defined, e.g.:

- Monitoring should be done in near-real time.
- In case of emergency staff should be notified automatically

- In case of serious accidents or fire alarms emergency services should be notified automatically
- Monitoring should detect or even predict problems in monitored systems or networks so that appropriate staff can take corrective actions to improve performance or prevent problems
- Information about detected problems should be provided to the other subsystems
- The video streaming function should have the possibility to prioritize a selected video stream
- The Remote Maintenance Subsystem should be able to get directives from other subsystems.

Yard Management Subsystem

Chapter 2.3.2 shows guiding principles on the VSP level that influence the yard management subsystem. Policies on the yard management subsystem level are:

- The Yard Management Subsystem should allow maximizing throughput of goods with a decrease of the error rate during scheduling.
- The Yard Management Subsystem should allow optimized flow of information for better transparency and analysability of processes on the yard.
- The distribution of notifications and the monitoring of the state of yard entities should be done nearly in real-time.
- A smooth loading or unloading process should be guaranteed, e.g. by advanced shipping notices.
- The Yard Management Subsystem should know the position of all Yard Jockeys.
- The assignment of Yard Jockeys to tasks should be done in an intelligent and efficient way, e.g. based on their location and further schedule.
- Information exchange between the Yard Management Subsystem and external organizations should be possible via Electronic Data Interchange.

Integration

A goal in the context of INDENICA is organization-wide reuse of services to reduce development costs. In order to reach this goal a Virtual Service Platform is introduced.

Appropriate policies have to be defined to track these goals. Examples for such policies on the VSP level also influencing the level of the existing platforms are already shown in chapter 2.3.2.

Additionally there are policies only relevant for the VSP level, e.g.:

- Services of the Virtual Service Platform shall be used preferably. Direct use of services of the underlying platforms is only allowed when services are not available through the Virtual Service Platform.

- For the user of VSP applications the VSP is a black box. The services of the underlying platforms are not visible for the user.
- Another high-level policy addresses the smooth operation of the integrated system. It is refined by a number of policies which include:
    - The remote maintenance video stream is prioritized when the warehouse system or the yard management system is in error state
    - The video stream of the yard reception is prioritized while a truck is doing reception process (Yard management <-> Remote maintenance)
    - The warehouse management gives goods storage process a higher priority, when there are too many delivering trucks on the yard.
    - The warehouse management gives the goods retrieval process a higher priority, when there are too many empty trucks on the yard.

4 Concept for INDENICA Platform Governance

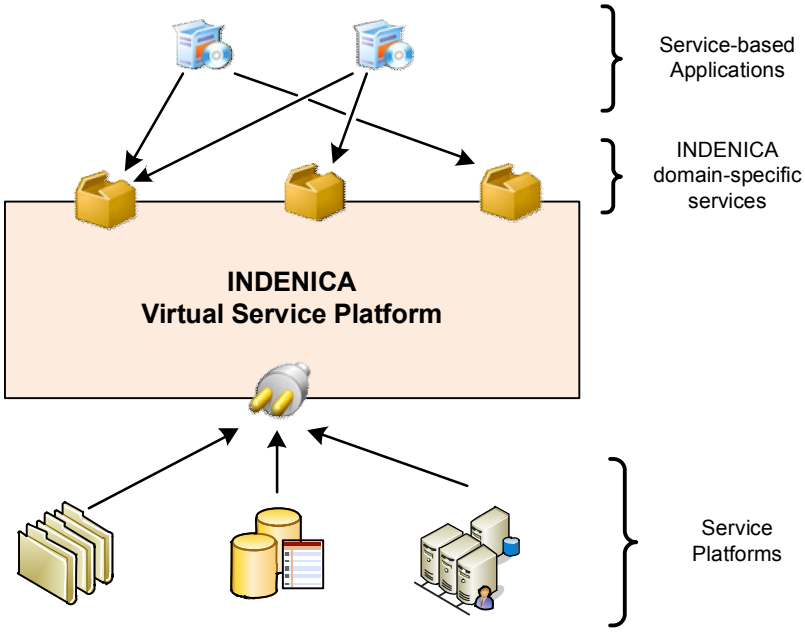

Figure 9: Overview of an INDENICA Virtual Service Platform (from Deliverable 3.1)

For the INDENICA Service Governance Framework the following VSP principles are relevant for the conceptual work and architecture governance:

- There are three layers of architectural and implementation work:
  - Applications
  - The Virtual Service Platform
  - The Base Service Platforms
- The application layer can be influenced by BPM Governance, IT Governance and SOA Governance
- The Virtual Service Platform is mainly driven by the SOA Governance, which in return has to take regard to specific aspects of variability
- The Base Service Platforms can be under the regime of distinct SOA governance depending on the ownership. A number of different legal constructions can be in place:
  - In-sourcing,
  - Out-sourcing,
  - Subcontracting

In chapter 2.3 we analysed the SOA Governance Framework elaborated by The Open Group and its degree of fulfilment with the requirements on governance by Khusidman.

Even if this SGF has some deficiencies (as the recent version is still classified “draft”) it provides a coherent overall structure and a set of terms and definitions that allows the development of the INDENICA Governance Framework as an extension to a generic SOA Governance Framework as shown in Figure 10.

In the ecosystem of governances INDENICA Governance can be seen as an extension to a generic Architecture Governance as specified in chapter 2.1.2.

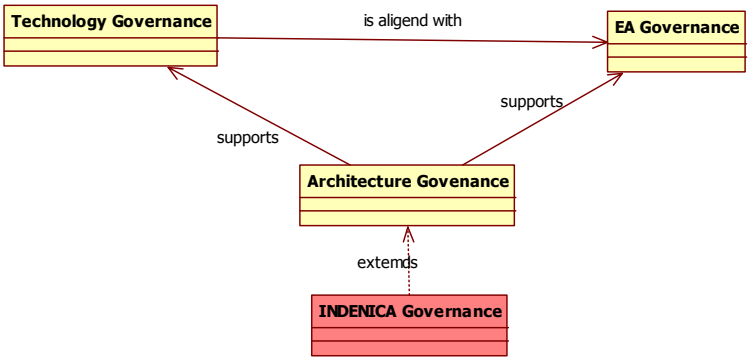

Figure 10: Position of the INDENICA Governance

The main structure of the INDENICA SGF will be the same as in the Open Group SGF:

- INDENICA Platform Governance Model as a tailoring of the SOA Governance Reference Model
- INDENICA Improvement Cycle as an extension to the SOA Governance Vitality Method

For the INDENICA Platform Governance Model we keep the structure and will add detailed analysis of

- Roles and responsibilities
- Governed processes
    - Platform Portfolio Management
    - Platform Lifecycle Management
    - Platform Development
    - Platform Change Management
- Policies and derived KPIs
- Monitoring and adaptation rules

Role-based Governance

The INDENICA overall architecture (see [INDENICA D3.1]) allows two main views on the main project's results:

- Role view
- Tool view

A generic task-based process meta-model (presented in Figure 11) defines the entities of a process and their relationships. Phases cover 2 or more milestones and end at only one. These phases also cover tasks which need input and output, thus are called results. Methods and tools are used by tasks to perform the work on results.

The relationship of roles to tasks are "responsible" or "participate"; in chapter 5.1 we will introduce a finer model that also contains relationships like "accountable", "consulted" or "informed".

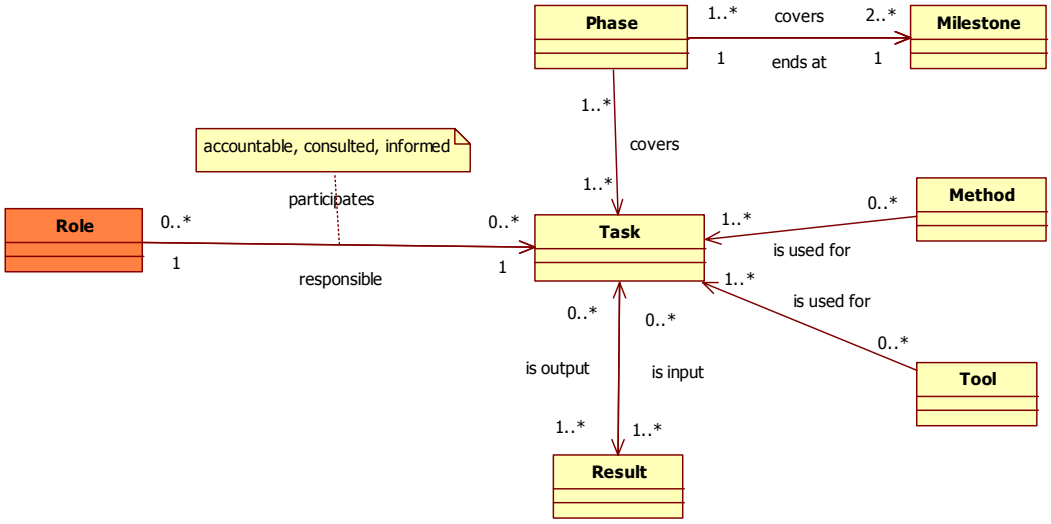

Figure 11: Process Meta Model

Tools support roles when they perform a task and produce results and this aspect is the motivation to take role definitions as he starting point of designing the INDENICA Platform Governance.

5 INDENICA Platform Governance Model

5.1 Role Model

In this chapter we set the focus on INDENICA specific roles and how they contribute to development and to governance activities. We merge INDENICA specific roles with roles and teams from the SOA Governance Framework to find out the additional elements which INDENICA could add to the framework.

5.1.1 Roles and Teams from the SOA-Governance Framework

In Table 1 the list roles of the SGF was structured in teams that should be in place in the context of SOA. The SGF covers the complete lifecycle of services, not only their development. This is also relevant for INDENICA because the project work also covers design time and run time aspects.

The Open Group SGF defines a comprehensive set of specific roles, especially different architects for the specific disciplines. But the framework is only in a draft status and role properties like “responsibilities” and “skills” are not defined for the listed roles. To merge the framework roles with INDENICA roles, the right level of authorisation has to be found. To ease the integration of the INDENICA roles, roles from the framework are clustered into categories according to their function in relation to the SOA program and to the enterprise (see Figure 12).

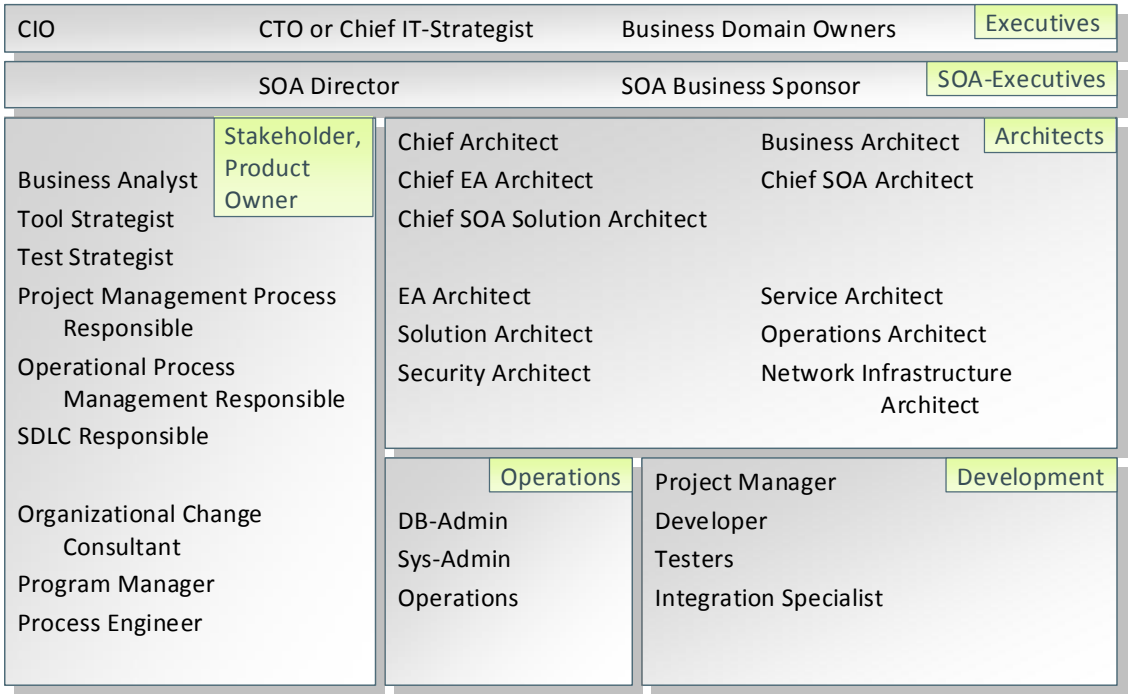

Figure 12: Clustered SGF Roles

Executives’ category: This category includes the upper management level of an enterprise.

SOA-Executives Category: This category includes the management level of the SOA programs of the enterprise.

Architects Category: This category includes two levels of architects, chief architects and architects. The chief architects are experienced architects from a certain domain, often leading a team of architects from the same domain. Architects are working in cross functional teams; chief architects are representing architectural aspects in cross functional teams on strategic level.

Development Category: This category includes all disciplines that are necessary for development of a service.

Stakeholder / Product Owner Category: This category includes the stakeholders and Product Owners of the services.

Operations Category: This category includes all disciplines needed to run the services.

5.1.2 INDENICA specific roles

5.1.2.1 INDENICA Roles Overview

This section first provides a high level introduction into the INDENICA roles. Next these roles are positioned in the SGF and the SGF-Teams. Afterwards, the roles will be described with their responsibilities, skills, rights and duties.

In chapter 3.2 we proposed six specific roles that are considered specific for development, deployment and usage of a virtual service platform:

- Platform Architect
- Platform Integrator
- Platform Provider
- Platform Variant Creator
- Platform Administrator
- Application Developer

As an extension to the overall architecture view in [INDENICA D3.1] we would like to define roles according to the principle to separate conceptual work ("Define") from implementation work ("Generate", "Deploy") and from the administration work at runtime (“Monitor”). This leads to a clearer distinction of responsibilities and skill profiles. Figure 13 shows an ideal development cycle for a VSP.

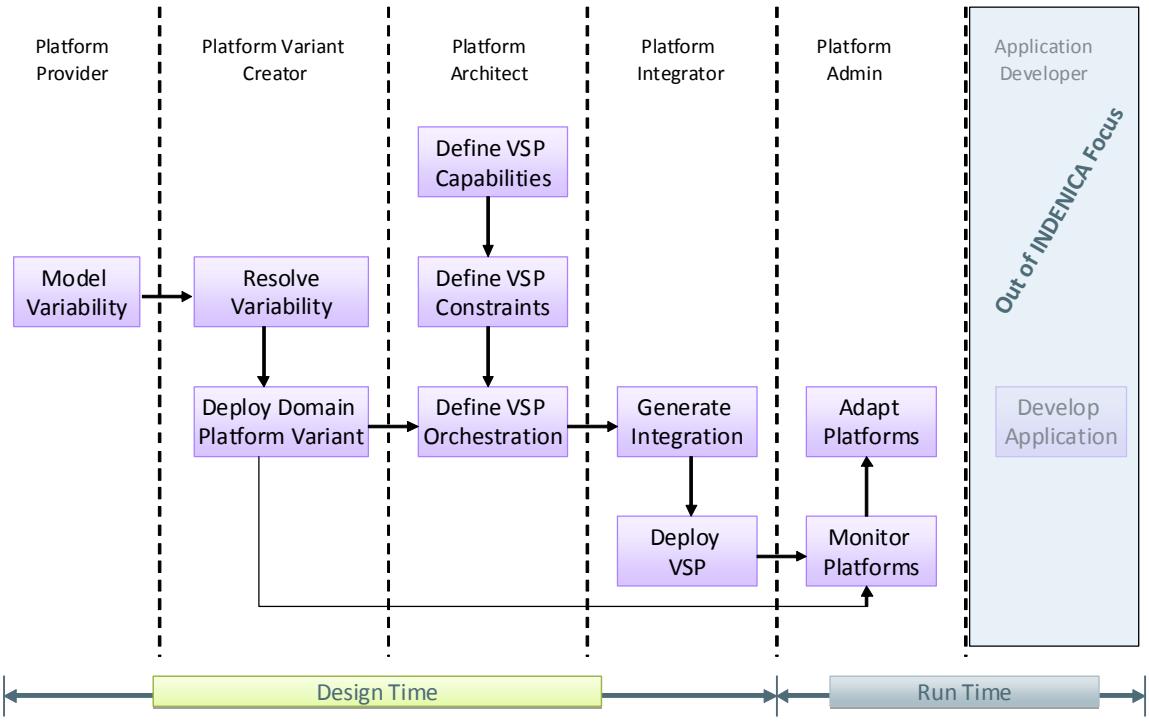

Figure 13: VSP Development Process Activities and corresponding roles

In reality, besides just configuring base platforms and the VSP, additional coding will be necessary, e.g. to implement the VSP specific features. If the base and domain platforms are developed by the enterprise itself, also these platforms have a development cycle. In simple words, each of the three platform included into the VSP development has the full set of development roles assigned to it (architect, developer, tester, integrator). Specific INDENICA activities are assigned to these existing roles. If the roles are distinguished by the type of platform development they belong to, the following mapping could be defined:

Platform provider	=	Base platform architect
Platform variant creator	=	Domain platform integrator
Platform architect	=	VSP architect
Platform integrator	=	VSP integrator
Platform admin	=	VSP admin

This is important as it will help to assign the work to be done in VSP projects to the right roles; especially when not all three platform development teams exist (e.g. the platforms are COTS products).

5.1.2.2 Mapping INDENICA Roles to SGF

This section presents a mapping of INDENICA roles to the SGF. Figure 14 shows the categories they fit in.

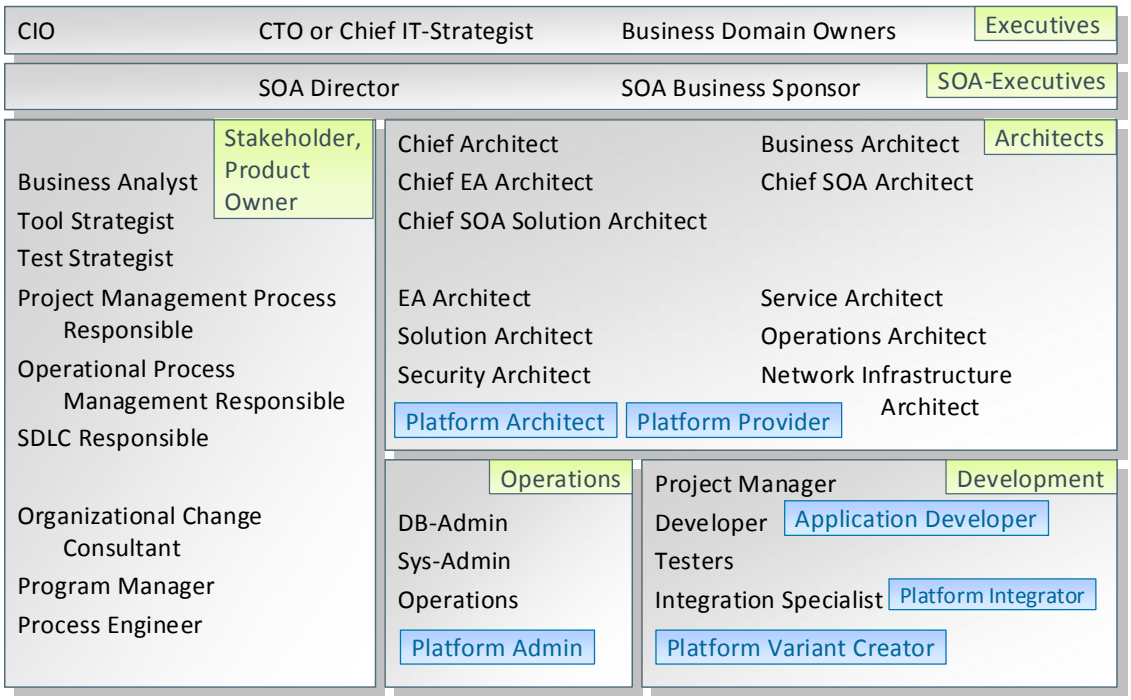

Figure 14: Mapping INDENICA Roles to SGF

Figure 15 to Figure 17 show the teams the INDENICA roles are assigned to.

(The roles participating in the teams are written in ***bold and italic*** letters.)

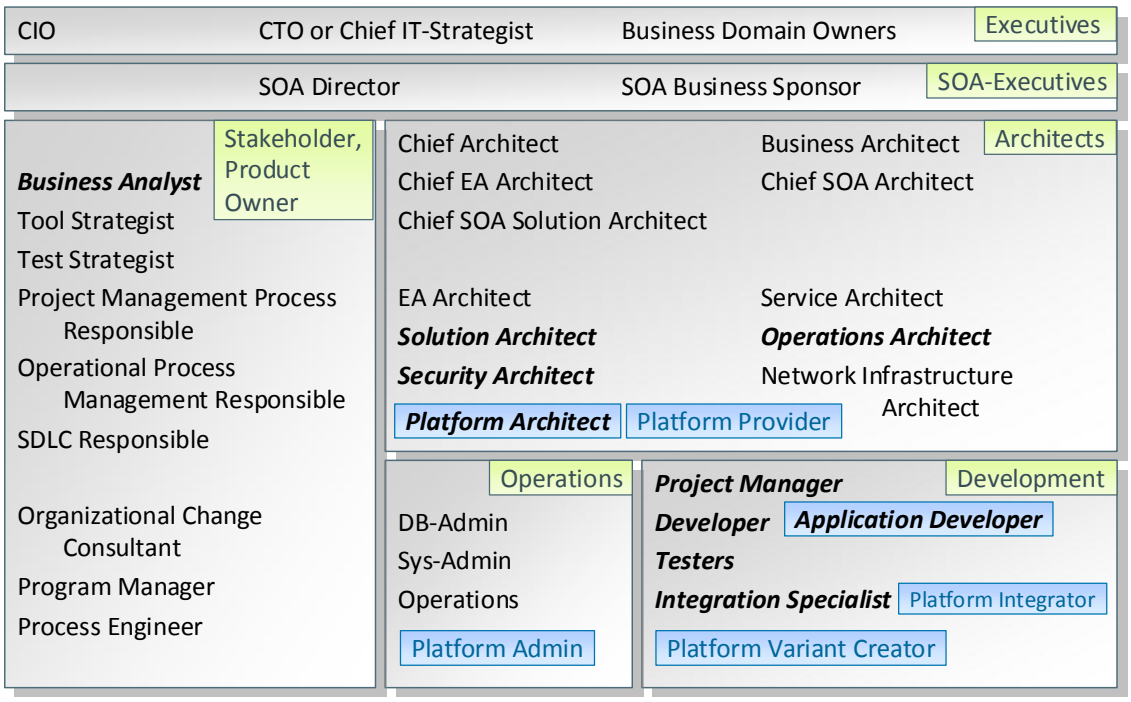

Figure 15: Solution Development Team: Execution and Delivery

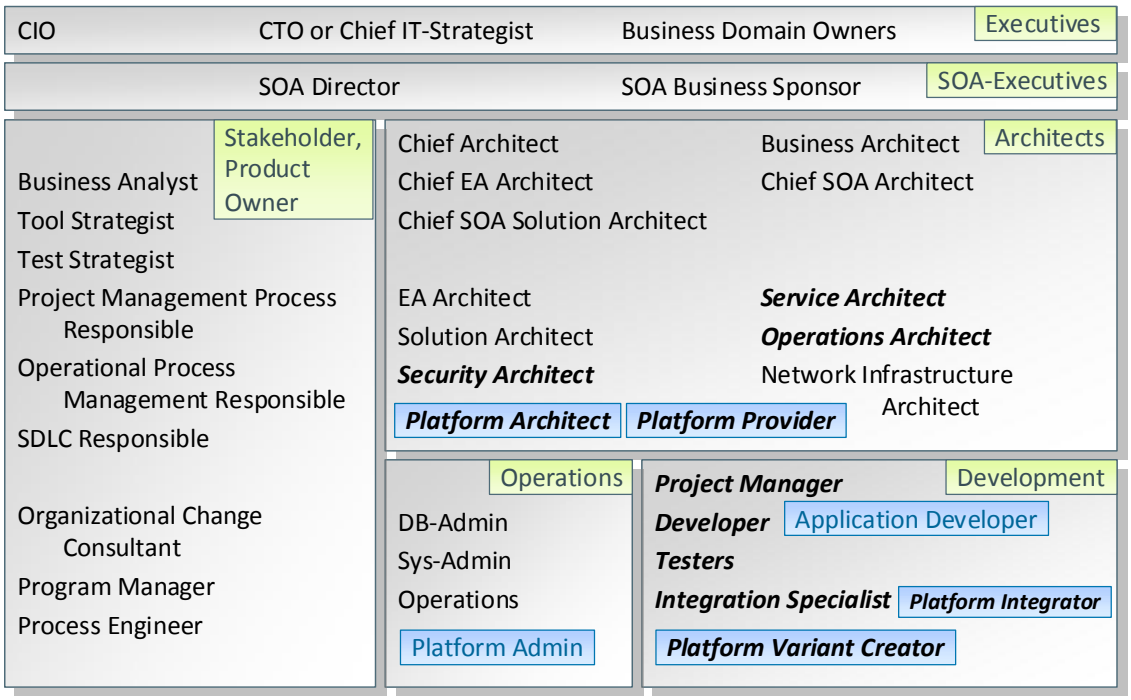

Figure 16: Service Development Team: Execution and Delivery

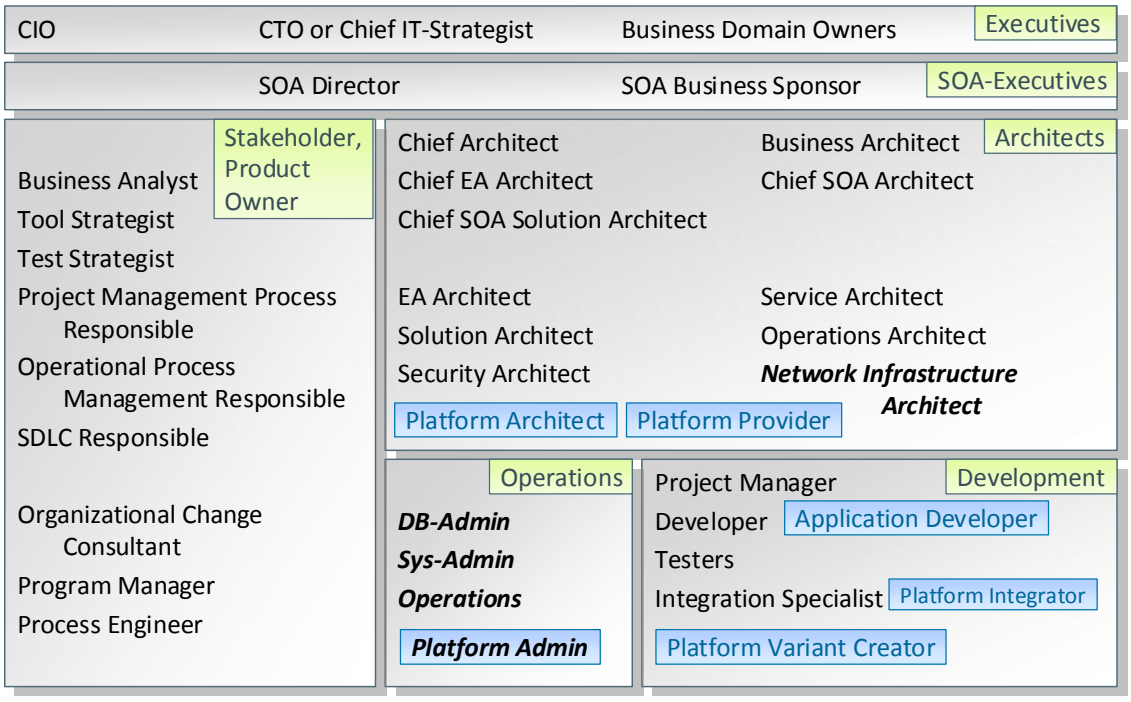

Figure 17: IT-Operations: Execution and Delivery

As expected, the INDENICA specific roles are located in teams close to development. Most of them are a specialized variant of a general role (e.g. application developer). To get more insight in those specialized roles we describe these in more detail in the next section

5.1.2.3 INDENICA Roles in Detail

In this chapter the six INDENICA roles are described in more detail. Skills, responsibilities, rights and duties will be defined. For the responsibilities part, a RACI model will be used. RACI is an acronym derived from the four key responsibilities most typically used: Responsible, Accountable, Consulted, and Informed. Different from the RACI model used by The Open Group SGF, we use an alternative RACI scheme from Wikipedia [Wikipedia 3]:

*"**R** – Responsible –*	*Those responsible for the performance of the task. There should be exactly one person with this assignment for each task.*
***A** – Assists –*	*Those who assist completion of the task.*
***C** – Consulted –*	*Those whose opinions are sought; and with whom there is two-way communication.*
***I** - Informed –*	*Those who are kept up-to-date on progress; and with whom there is one-way communication.*"

This definition of responsible and assists are well suited for the activities covered by INDENICA. RACI is good for describing interdisciplinary work. Instead of activities also work products from the product breakdown structure could be listed. The result could also be published in a responsibility assignment matrix, which shows the participation of different roles in activities or work products.

In the following tables, the INDENICA roles are described in more detail. The role description contains three parts:

- Activities: All activities which the role is involved in are listed according to the classification of the RACI model. This could be development or governance relevant activities
- Rights and Duties: Rules to follow are listed here together with the entitlements.
- Skills and Competencies: All knowledge and skills needed to fulfil the role tasks are listed here.

Platform Provider	
Activities	Responsible: ▪ for variability modelling (describing the current variability and the variability binding process of the base platform he owns) Assists: ▪ in analysing change requests ▪ in implementing base platform relevant change requests
Rights and Duties	▪ Act compliant to the valid governance policies ▪ Participate in CCB

Skills and Competencies	▪ Technology expert for base platform ▪ Knowledge of the existing variability and capabilities of a base platform ▪ Knowledge about variability models and variability modelling techniques ▪ Conceptual and abstract thinking ▪ Working in cross functional teams

Platform Variant Creator	
Activities	Responsible: ▪ for binding unresolved variability in base platform(s) ▪ for creating an executable platform variant optional[4]: ▪ for specifying additional functionality not covered by the base platform ▪ for variability modelling of the additional functionality ▪ for implementing the additional functionality Assists: ▪ in analysing change requests ▪ in implementing domain platform relevant change requests
Rights and Duties	▪ Act compliant to the valid governance policies ▪ Act compliant to the design time policies for domain platforms by binding (and implementing) appropriate functionality ▪ Participate on request in CCB
Skills and Competencies	▪ Is a technology expert for domain specific platform ▪ Knowledge of one or more existing (possibly domain specific) platforms ▪ Knowledge about the variability models and their implementation in the platform

Platform Architect	
Activities	Responsible: ▪ for designing VSP Capabilities (requirements management), ▪ for defining the variability within VSP, ▪ for defining VSP constraints, ▪ for defining VSP orchestration, ▪ for creating the baseline architecture and ▪ for creating the baseline adaptation behaviour of VSP ▪ for analysing change requests Assists:

[4] Ideally there should be no additional functionality needed. Real life is different. If the domain platform is developed in-house, there may be architects and developers responsible for the development. So they would be in charge for these activities. Otherwise the Platform Variant Creator could be responsible.

	▪ in implementing VSP relevant change requests ▪ in defining KPIs and rules
Rights and Duties	▪ Act compliant to the valid governance policies ▪ Act compliant to the design time policies for VSP by binding and implementing appropriate functionality ▪ Act compliant to the design time policies for domain platforms by selecting appropriate platforms for integration ▪ Participate in CCB
Skills and Competencies	▪ Is a technology expert for VSP ▪ Works in interdisciplinary teams with domain platform experts and application experts ▪ Conceptual and abstract thinking ▪ Knowledge about requirements engineering methods and techniques ▪ Knowledge about architecture definition ▪ Knowledge of all domain specific platforms to be integrated into the VSP ▪ Knowledge about the variability models and their implementation in the VSP

Platform Integrator	
Activities	Responsible: ▪ for generating the integration of the domain platforms to the VSP ▪ for generating the executable VSP instance (->Deploy) ▪ for implementing KPIs and rules Assists: ▪ in solving VSP relevant change requests ▪ in implementing VSP relevant change requests ▪ in defining KPIs and rules
Rights and Duties	▪ Act compliant to the valid governance policies ▪ Act compliant to the run time policies for domain platforms and VSP by implementing monitoring rules according to the KPIs belonging to the run time policies. ▪ Participate on request in CCB
Skills and Competencies	▪ Knowledge of all domain specific platforms to be integrated into the VSP ▪ Work in interdisciplinary teams with domain platform experts and application experts ▪ Conceptual and abstract thinking ▪ Knowledge about architecture definition

Platform Administrator	
Activities	Responsible: ▪ for monitoring the current state of the domain platform instances and the VSP instance (via KPIs and rules) ▪ for making adaptations of the VDP instance
Rights and Duties	▪ Act compliant to the valid governance policies ▪ Adapt the VSP and the domain platforms in the frame of the given rules ▪ Inform Platform Architect if platform adaptations are not compliant to policies and ask for dispensation
Skills and Competencies	▪ Technology expert for VSP (for reconfiguring the platforms) ▪ Also technology expert for domain platforms (for reconfiguring the platforms)

Application Developer	
Activities	Responsible: ▪ For developing applications based on the VDP instance ▪ For implementing the additional business logic specific for the application Assists: ▪ in analysing change requests ▪ in solving application relevant change requests for VSP
Rights and Duties	▪ Acts compliant to the valid governance policies ▪ Uses services provided by VSP ▪ Applies for dispensation if services covered by the VSP are not used, but other external services ▪ Does not modify the VSP ▪ Participate in CCB
Skills and Competencies	▪ Knowledge about services and usage of services provided by VSP ▪ Knowledge about business logic specific for the application

With this INDENICA role model we have a comprehensive definition of specific roles and their allocation to the context of a SOA Governance Framework.

In the following chapters we will look at the involvement of these roles in governed processes.

5.2 Governed Processes

While some high level processes do not address specific aspects of a Virtual Service Platform, the Portfolio Management and the Lifecycle Management Processes will have to cope with different life cycles of base platforms and the virtual platform.

In addition to these there will be implications on the development and the operation of Virtual Service Platforms. Here we will analyse in detail the Architecture Governance, the suitability of iterative development approaches and the Service / Platform change management

These processes will be described in Detail in chapter 5.2.1.

The Open Group [SGF 2009] describes the relationship of Lifecycle and Portfolio Management as: *"The Solution Portfolio Management process focuses on planning and prioritization of individual SOA solutions. These individual solutions may consume existing services as well as define new services. Following the guidance of Service Portfolio Management process, these solutions may consume the reusable services developed by Service Lifecycle process and/or define new services for Service Portfolio Management. The new services are thereby prioritized by Service Portfolio Management for the Service Lifecycle process to manage for consumption by the individual SOA solutions. The Solution Lifecycle then enforces the Solution Portfolio Management plans during the development, deployment and management of the individual SOA solution. "(*See also Figure 18)

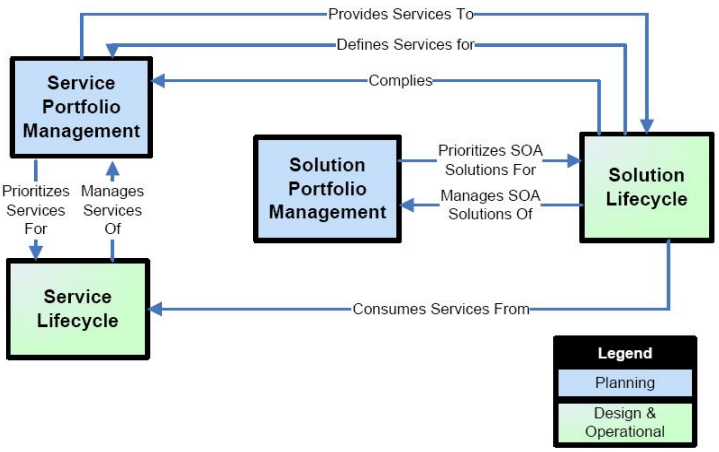

Figure 18: Governed SOA Process Relationships (from [SGF 2009] p. 23)

When introducing the concept of a VSP we have to consider that the base platforms can have different owners, all of which have their service portfolios and roadmaps. A decision to use a service platform always must be based on - among others - the analysis of these portfolios and roadmaps and their ability to serve the solution portfolio.

5.2.1 Platform Portfolio Management

A platform portfolio is a document (be it paper or digital) that describes from an IT and EA strategy point the platforms an enterprise has in use or plans to use. As a basis for implementation planning the portfolio is mapped to a timeline, called platform roadmap.

In analogy to a Product Portfolio Management Processes and Platform Portfolio Management Process consists of four phases.

Figure 19: Platform Portfolio Process

The trigger for any changes of the platform portfolio is the regular update of the solution- or application portfolio. A thorough and methodical analysis of the solution portfolio unveils all needs for services and platforms.

These identified needs and change proposals need to be mirrored at the capacities of the existing platform and service portfolio and identified gaps these portfolios have to be analysed in technological and business aspects.

In case of introducing a VSP, once the first decisions on the scope of platforms are in place, the scope of the virtual service platform is defined using requirements models described in [INDENICA D1.2.1] . The scoping decisions on this level can be supported by ROI calculations as also defined in this deliverable.

The decisions to add new services to the portfolio and to redraw existing services are then documented in the new release of the platform portfolio. This sets the goals for a certain period of time – usually 2 to 3 years and shall be in line with the solution portfolio.

Finally a new version of the portfolio is mapped on a timeline and displayed as roadmap. It has to be reviewed by all stakeholders and affected parties and has to be aligned with the budget planning.

The four phases contain in detail:

Analyse Solution Roadmap

Phase	Analyse Solution Roadmap
Goal	Identify needed changes to service and platform portfolio
Input	New service needs Service change proposals
Tasks	Manage new service needs received and analyze their business justification. Manage service change proposals. Manage new and changed service contracts.
Output	List of service needs and changes
Role(s)	EA Governance Board: Chief Enterprise Architect (R) Chief SOA Architect (A) Enterprise Architect (A)

Evaluate Platform Portfolio

Process	Evaluate Platform Portfolio
Goal	Review existing portfolio of platforms and services and get insight of its capacities and capabilities to fulfil

Input	Service usage plan and contracts Service funding model
Tasks	Identify service capacity in relation to service usage needs from the service contracts and propose changes to the services if needed. Analyse emergent usage patterns to identify shortcomings or gaps in the current service portfolio.
Output	Service capacity and gap analysis
Role(s)	EA Governance Board: Chief Enterprise Architect (I) Chief SOA Architect (R) Enterprise Architect (A)

Develop Platform Portfolio

Phase	Develop Platform Portfolio
Goal	Decide on new services, changes to existing services, refactoring of existing services, and retiring of existing services.
Input	List of service needs and changes Service capacity and gap analysis
Tasks	Assign ownership to new services. Catalogue the services that will be created, enhanced, used, or retired as part of the projects that implement the service roadmap.
Output	Platform Portfolio, Service Catalogue
Role(s)	EA Governance Board: Chief Enterprise Architect (C) Chief SOA Architect (R) Enterprise Architect (A)

Develop Platform Roadmap

Phase	Develop Platform Roadmap
Goal	Put platform portfolio and service catalogue on a timeline regarding solute priorities and balancing with development capacities
Input	Platform Portfolio, Service Catalogue
Tasks	Decide on implementation priorities and budget
Output	Platform and service roadmap New overview service descriptions including • service contracts for the first consumers • service policy • service classification

	• service ownership • business justification • usage plan (if the service will use other services)
Role(s)	EA Governance Board: Chief Enterprise Architect (C) Chief SOA Architect (R) Enterprise Architect (A)

5.2.2 Platform Lifecycle Management

The Lifecycle Management of a Virtual Service Platform faces the challenge that it is not only influenced by the stakeholder and their needs, but also the underlying domain platforms usually have their own lifecycles. A strong coordination between these concurrent lifecycles is necessary.

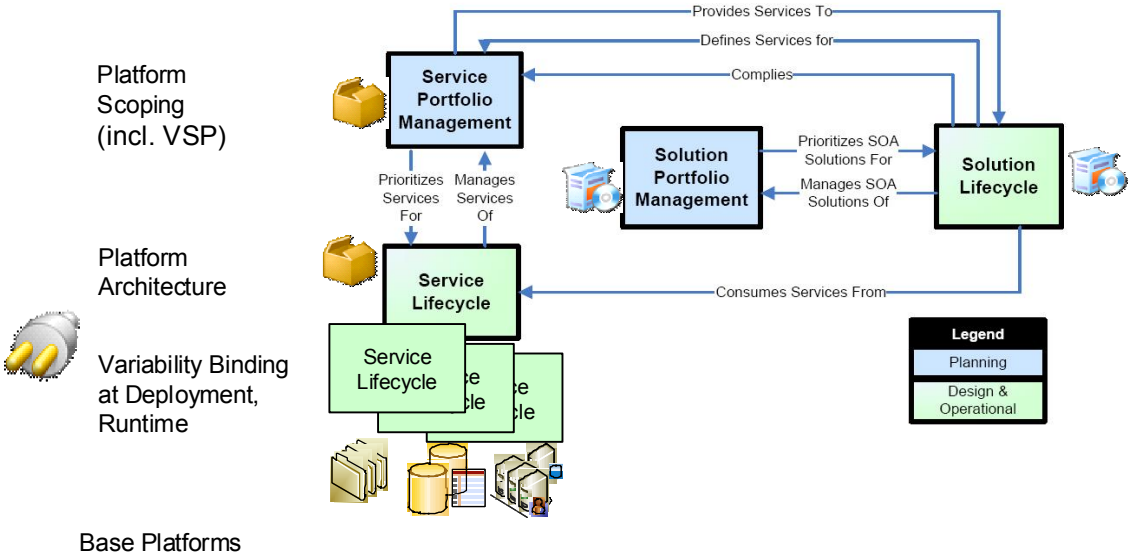

Figure 20: Platform Portfolio Management Context (based on [SGF 2009])

Seen from the viewpoint an enterprise there can only be one service portfolio which has to take into account that the services have different lifecycles. I case of a VSP and thus external platform ownership some of the lifecycles may not be under direct control and responsibility. Figure 20 gives an overview on this situation:

- The Service Portfolio Management decides on platforms to be used and scoping of a VSP
- The Service Lifecycle for a VSP services defines a “filtered” view of services exposed by the VSP
- The Service Lifecycles for Base Platform Services provide the view on services from the viewpoint of the platform owners

Mechanisms for aligning these service lifecycle views are described in chapter 5.2.2. Here we will go further into the detail of the portfolio management.

A Platform Lifecycle Management Process can be derived from a standard product or solution process with some adaptations as shown in Figure 21.

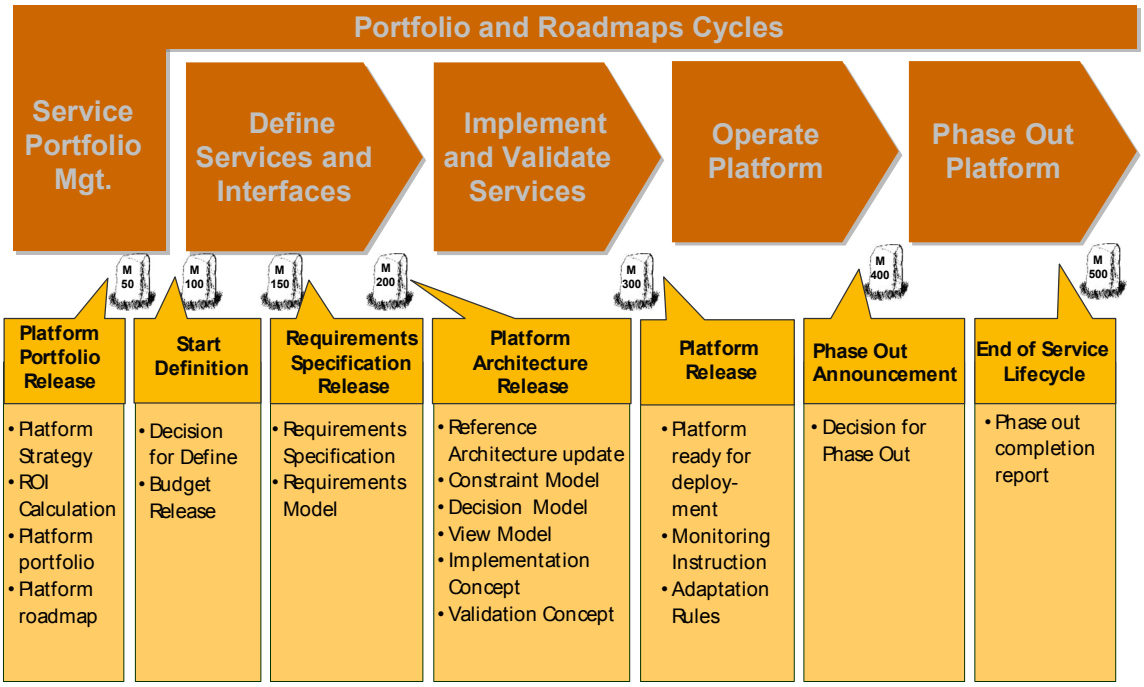

Figure 21: Platform Lifecycle Process

The four lifecycle phases are Define, Implement, Operate and Phase Out. They are separated by specific milestones with two additional ones at Portfolio Management and within the Define phase.

M50 is the release point for the platform portfolio. It occurs on a regular basis with duration depending on the business. Usually the portfolio is released on a yearly basis. Interim releases can be necessary, when triggered by major changes in business strategy.

M100 is the starting point for a specific platform development and does not necessarily coincide with M50. The roadmap specifies the order of starting points which depend on the size of prospected projects, target release dates and development capacities of the organization.

M150 is a checkpoint within the Define phase where the requirements are complete and a commitment is obtained on the feasibility and implementation budget. In case of a Virtual Service Platform it shall also include the requirements models resulting from INDENICA tools.

M200 is the release of the architecture the implementation and the validation concept. In case of a Virtual Service Platform it shall also contain all models resulting from INDENICA tools.

M300 is the release of the platform ready for deployment and validated according to the validation concept. With this milestone the operation of the platform begins and monitoring and adaptation is performed according to the respective rules. Changes to the platform shall be governed by the change management process.

M400 is the starting point to phase out the platform. This can be triggered by business needs or major technological changes. In any case the decision is done during a portfolio management cycle and has to be part of a portfolio update (M50).

M500 is the end of all operation, the platform is removed, all service contracts are closed and a replacement strategy is implemented.

Note:

The presented Platform Lifecycle Management Process sets a generic frame that has to be filled with concrete detailed processes for each phase. In chapter 5.2.4 we will describe the implementation of Agile Development of a platform as an example policy which covers the phases Define and Implement of the lifecycle process.

In the context of INDENICA there is an additional challenge resulting from the fact that base platforms and the VSP are developed in parallel. Seen from a single platform the need for synchronizing is not obvious, but for implementing a VSP it is indispensable to have a sync mechanism in place. Figure 22 shows the dependencies and needs for synchronization of the lifecycles. Required synchronisation points are:

Synchronisation Point 1: Requirements

At this synchronisation point the requirements models of base platforms and the VSP must be aligned. This can be achieved by following means:

- Participation of the Platform Provider and Platform Variant Creator in VSP prioritization and decision meetings
- Mutual review of requirements and decision models and specifications
- Agreement on aligned requirements and decision models and specifications

Synchronisation Point 2: Architecture

At this synchronisation point the architecture of base platforms and the VSP must be aligned. This can be achieved by following means:

- Participation of the Platform Provider and Platform Variant Creator in VSP architectural decision meetings
- Mutual review of platform capabilities, constraints and orchestration
- Agreement on a common reference architecture and individual design models and specifications

Synchronisation Point 3: Implementation

At this synchronisation point the integrated a tested base platforms and the VSP must be aligned. This can be achieved by following means:

- Participation of the Platform Provider, Platform Variant Creator and Platform Architect in VSP integration and release meetings
- Performance of integrated test suites for base platform variants and VSP
- Agreement on common release of base platforms and VSP for deployment

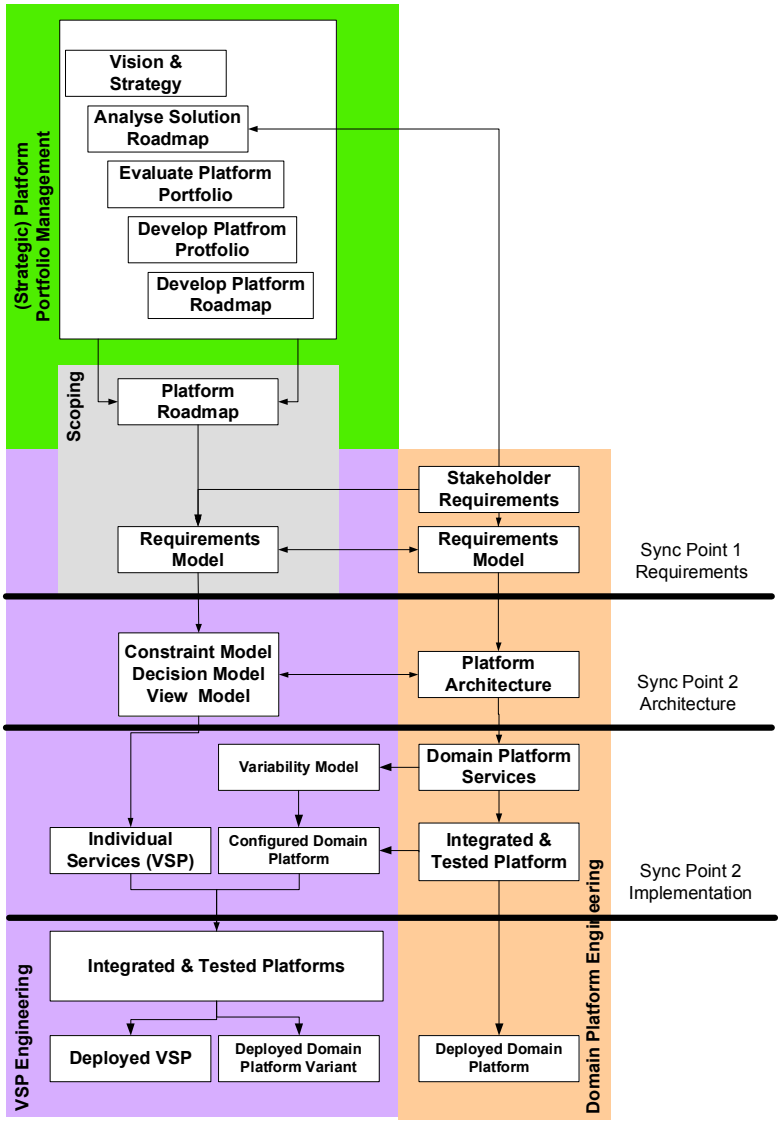

Figure 22: Synchronisation of Platform Lifecycles

5.2.3 Platform Architecture Governance

The INDENICA specific roles are all related to one of three skill and competence groups: architects, development and operations. This triggers to have a closer look at the detailed processes and policies on this level.

In chapter 3.3 we listed a number of policies that relate to the implementation of base platforms and the VSP. Some of these policies relate to runtime behaviour and will be further analysed in chapter 5.3.

The policies relevant for design time are a significant part of the Architecture Governance, while those concerning run time can be directly processed as input for the requirements model.

All these policies can only be implemented and controlled by a mature development process with defined roles, tasks and results. It shall contain among others:

- Establish an architecture control board
- Introduce owners of the VSP
- Provide policies and guidelines for modification and evolution of base platforms and the VSP
- Provide policies and guidelines for compliance with standards and regulations
- Provide policies for the usage of Open Source software and investigation of intellectual property rights
- Provide tools for checking and enforcing policies, tools to asses architectures and test suites

In the INDENICA Governance the architecture control board is covered by the SOA Governance Board. It shall be informed on all architectural decisions and ensure compliance with policies, standards and guidelines. The owners of the platforms are the Platform Provider and Platform Architect roles.

An example for guidelines for compliance with standards and regulations addressing agile development and regulated environments is given in chapter 5.2.4.

5.2.4 Guideline for Agile Development in Regulated Environments

In this chapter we will introduce a sample guideline for developing platforms under two principles:

a) Apply an agile method during Define and Implement phase

b) Take strong regard to regulations that are applicable in the domain.

We will outline SCRUM as the most widespread agile development method in industry and regulations that are valid in the business areas of Siemens.

The detailed guideline is then attached in the Annex.

As the agile method we use SCRUM, which follows the Agile Manifesto [Agile 2012] and gives communication a higher priority than documentation. But the application of SCRUM often conflicts with rules and regulations that the regulated business and technological area have to comply with, such as:

- FDA Regulations for medical devices
- CENELEC Norms for Electrical Engineering
- RTCA Do-178B for Avionic Software

All these regulations require (in conjunction with ISO 9001 and others) a written documentation of the development process and product classifications into levels of safety relevance.

In order to solve the contradiction between agile development and regulations the sample guidelines will help to follow the regulations and to produce all required documentation without sacrificing the benefits of agile development.

General Hints for applying an agile methodology in regulated environments are

- Emphasize the responsibility of the team that owns its processes/practices.
- Discipline is indispensable. This refers to:
    - the application of documentations, traceability documentation, coding standards, and the change management,
    - quality and project planning including length of iterations and validation points,
    - awareness of product safety.

Overview of a typical SCRUM process model

SCRUM is an iterative Process Model that has become a widespread standard for agile Software development in the industry [SCRUM 2012]. It allows applying an iterative approach combined with a set of agile management practices. The pure SCRUM approach identifies only three roles:

- **The Product Owner** represents the customer and all other stakeholders and is responsible for specifying the product's requirements, prioritizing and ordering them in product backlog, and for accepting / rejecting the product increments.
- **The Team** is responsible for all development, integration and testing tasks and shall commit to deliver the planned content of an iteration called "sprint". A sprint has a fixed time of about 3 to 6 weeks (recommended).
- **The SCRUM Master** is an expert in performing and managing SCRUM projects and shall motivate and guide the team in procedural matters to reach the sprint goals. He moderates the "Daily Scrum", a short (about 15 min) meeting of the team where progress and issues are discussed, the Sprint Review Meeting, Sprint Retrospective Meeting and the Sprint Planning Meeting.

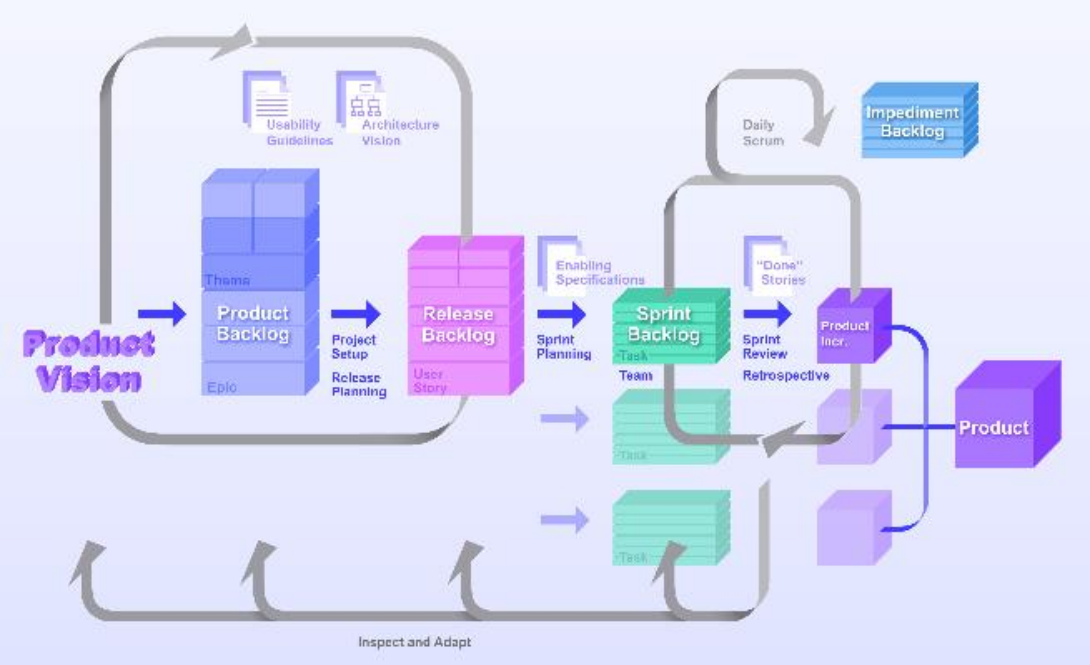

Figure 23 SCRUM Process Model

Te full text of the Guideline for agile development in regulated environments is attached in the Annex on page 68.

5.2.5 Service Change Management Process

Changes to platforms and services can occur at any time and for various reasons. As agile approaches help to decide on and implement changes by planning iterations and releases during specification, modelling and implementation, there are points in time that require a formal process for deciding on change requests and implementing them.
Such a Change Management Process is always applicable when work products are affected that have been approved by a formal decisions, e.g.:

- Reference Architecture
- Required Specifications according to regulations
- Released platform and services

In regulated environments a dedicated Change Management Process is required. It shall be applicable on following changes and contain respective actions:

- **Design changes**: Identify, document, validate/verify, review and approve design changes;
- **Document changes**: Review and approve by certain qualified and designed individual(s) document changes, communicate them, have change records (description of the change, identification, signature of approval incl. date, and date when the change becomes effective);

- **Process changes**: Establish procedures for changes to a specification, method, process, or procedure. Such changes shall be verified / validated, the process shall be evaluated & revalidated (if appropriate), the activities documented, and the change approved as above.

The Open Group SGF sees the Change Management Process as a part of the Service Portfolio Process and gives a number of reasons for applying it:

- newer versions and configurations of interfaces
- changes in the database structure
- operating system or network system update

"These application, infrastructure and environmental impacts are outside of the scope of classic configuration management tools. Appropriate change control processes have to be stringent and well devised to capture such a large scope, and are difficult to implement (and rarely are)."

A generic Change Management Process contains tasks as shown in Figure 24. The main decisions shall be taken by a board of business and technical experts. Usually in practice this is called Change Control Board (CCB). In [SGF 2009] consequently the CCB is covered by the owner of the portfolio which is the EA Board (see chapter 2.3.4).

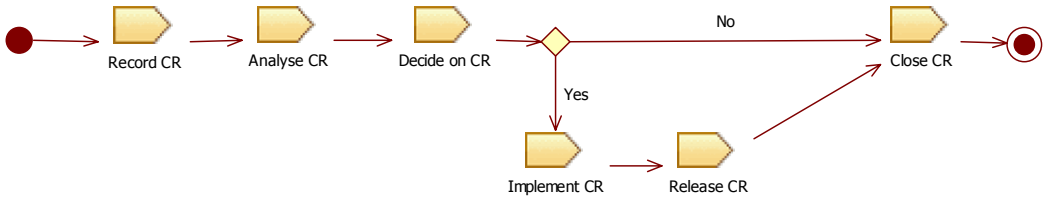

Figure 24: Change Management Process

In case of implementing a VSP, a Change Control Board is also required on a lower level for changes in the project or on changes that are technically triggered but do not affect the service and platform portfolio. In this case the CCB would be composed as shown in Figure 25. The EA Board is represented by a delegate.

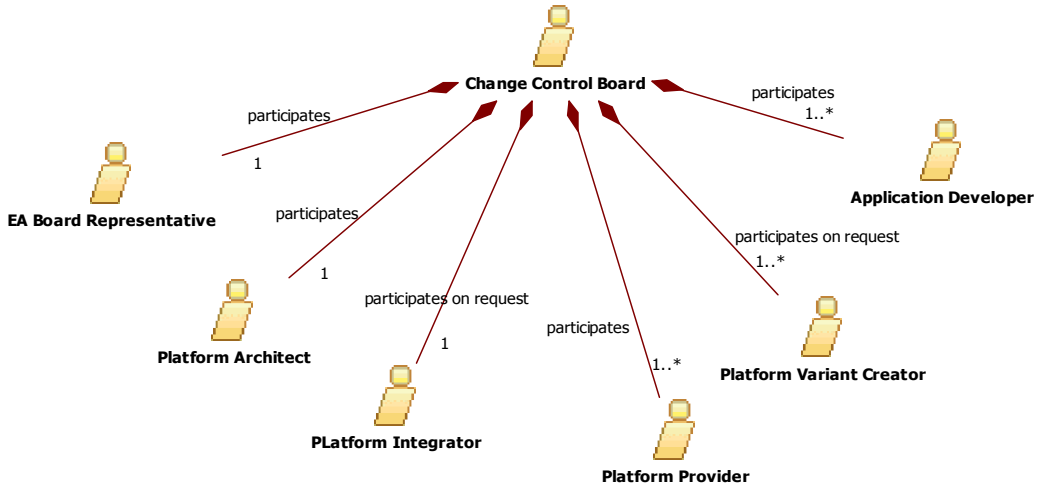

Figure 25: Change Control Board for VSP and Base Platforms

Thus the tasks of the Change Management Process would be defined as follows:

Task	Record Change Request
Goal	Change requests are registered in the change request data base
Input	Change requests submitted by mail, email, direct input into data base, …
Work Definition	• Enter CR into the data base • Allocate unique identifier • Pre-categorize CR • Allocate CR to CCB for analysis
Output	Pre-categorized CR in data base CR data base updated
Role(s)	CR data base administrator[5] (R)

Task	Analyse Change Request
Goal	CR is analysed and evaluated with clear statement on impact on portfolio, cost and benefit
Input	Pre-categorized CR in data base
Work Definition	• Evaluate impact on service portfolio • Evaluate impact on base platforms and VSP • Calculate prospected implementation costs • Evaluate benefit for solutions • Evaluate risks
Output	CR analysis report containing

[5] Additional generic role, empowered by CCB, not part of the SGF

	• Cost / benefit calculation • Risk evaluation • Decision proposal for CCB
Role(s)	Platform Architect (R) Platform Provider (A) Platform Variant Creator (A) Application Developer (A)

Task	Decide on Change Request
Goal	Decide on accepting or rejecting CR
Input	CR analysis report
Work Definition	• Discuss result of CR analysis • Decide on implementation or rejection
Output	CR Decision
Role(s)	CCB (R)

Task	Implement Change Request
Goal	Service changes, changes to VSP and Domain Platform are implemented and ready for deployment
Input	CR decision CR analysis report
Work Definition	• Update requirements, decision and view models • Generate code • Add manual code • Integrated and test service
Output	Service(s) implemented or changed (source code)
Role(s)	Developer (R) Platform Architect (A) Platform Provider (A) Platform Variant Creator (A) Platform Integrator (A)

Task	Release Change Request
Goal	Services are deployed and available
Input	Service(s) implemented or changed (source code)
Work Definition	• Deploy service on base platform or VSP • Update variability model • Update service contracts
Output	New or updated Service New platform versions CR implementation and deployment report
Role(s)	CCB (R)

Task	Close Change Request
Goal	CR closed
Input	CR decision CR implementation and deployment report
Work Definition	• Update CR database according to decision and (if applicable) report
Output	CR closed
Role(s)	CR Data Base Administrator (R)

5.3 KPIs for Governance of Virtual Platforms

In this section we apply the Goal-Question-Metric (GQM) software metrics approach to the high-level policies defined in Section 3.1.1 and 3.3. Starting from these policies our goal is to identify the measurable metrics that will allow us to determine, at runtime, if our systems are satisfying the policies. In other terms, our goal is to derive the set of Key Performance Indicators that we want to measure in our use cases. The Key Performance Indicators can then be translated into specific monitoring rules depending on the monitoring capabilities deployed with the system.

The following table shows an overview of the policy statements derived from the Case Studies. The IDs are built according to the following rule:

P_<subsystem/system><PolicyNr>

"P" stands for policy. "Subsystem/System" can be "WH" for Ware House, "RM" for Remote Monitoring, "YM" for Yard Management or "VSP" for Virtual Service Platform (respectively Integration).

The table also lists a classification of the policies, if they are enforced at run time or at design time.

ID	Policy Statement	Design Time	Run Time
P_WH1	The Warehouse Management Service utilisation must be registered at the warehouse platform owner.	x	
P_WH2	The usage of the transport control service is restricted to the warehouse internal applications and must be protected by appropriate authentication.	x	
P_WH3	The rejection rate of storage requests should be less than 1%.		x
P_WH4	Retrieval jobs have to be processed en-bloc.		x
P_WH5	The rate of non-retrieval of requested items shall not exceed 0.1%.		x
P_RM1	Monitoring should be done in near-real time.		x
P_RM2	In case of emergency staff should be notified automatically		x
P_RM3	In case of serious accidents or fire alarms emergency		x

ID	Policy Statement	Design Time	Run Time
	services should be notified automatically		
P_RM4	Monitoring should detect or even predict problems in monitored systems or networks so that appropriate staff can take corrective actions to improve performance or prevent problems		x
P_RM5	Information about detected problems should be provided to the other subsystems	x	
P_RM6	The video streaming function should have the possibility to prioritize a selected video stream	x	
P_RM7	The Remote Maintenance Subsystem should be able to get directives from other subsystems.	x	
P_YM1	The Yard Management Subsystem should allow maximizing throughput of goods with a decrease of the error rate during scheduling.		x
P_YM2	The Yard Management Subsystem should allow optimized flow of information for better transparency and analyzability of processes on the yard.	x	
P_YM3	The distribution of notifications and the monitoring of the state of yard entities should be done nearly in real-time.		x
P_YM4	A smooth loading or unloading process should be guaranteed, e.g. by advanced shipping notices.		x
P_YM5	The Yard Management Subsystem should know the position of all Yard Jockeys.		x
P_YM6	The assignment of Yard Jockeys to tasks should be done in an intelligent and efficient way, e.g. based on their location and further schedule.	x	
P_YM7	Information exchange between the Yard Management Subsystem and external organizations should be possible via Electronic Data Interchange.	x	
P_VSP1	Services of the Virtual Service Platform shall be used preferably. Direct use of services of the underlying platforms is only allowed when services are not available through the Virtual Service Platform.	x	
P_VSP2	For the user of VSP applications VSP is a black box. The usage of services of the existing platforms is not visible for him.	x	
P_VSP3	The remote maintenance video stream is prioritized when the warehouse system or the yard management system is in error state		x
P_VSP4	The video stream of the yard reception is prioritized while a truck is doing reception process (Yard management <-> Remote maintenance)		x
P_VSP5	The warehouse management gives goods storage process a higher priority, when there are too many delivering trucks on the yard.		x
P_VSP6	The warehouse management gives the goods retrieval process a higher priority, when there are too many empty trucks on the yard.		x

Table 2 Policies Overview and Classification

These policies are the basis for the further work with GQM.

GQM is a model composed of three levels. First we have a conceptual level in which the analyst defines the high-level goals that the system needs to reach. In this section we assume that this first level has been completely defined in section 3.1.4 with the high-level policies. Second we have an operational level in which the analyst defines a set of questions that better define the high-level goals as completely as possible. Finally, we have a quantitative level in which the analyst defines a set of metrics for each question so that it can be answered in a measurable way.

In the following subsections we will report the results of the GQM analysis performed for the three subsystems identified in the use cases, as well as for their integration. Notice however that not all the policies listed in Table 2 can be effectively translated into KPIs that can be later monitored. Indeed, some of the high-level policies are intrinsically satisfied at design time and for the following sections we will only focus on run time policies and KPIs. The following KPIs are defined using configurable numeric thresholds. We have provided reasonable examples for these thresholds in order to make the exposition of the KPIs more meaningful to the reader. Never the less, they may be further changed during implementation of the case studies by domain-experts.

A unique ID identifies all the goals and KPIs. The IDs are built according to the following rule:

G_<system/subsystem><PolicyNr>_<GoalNr>

KPI_<subsystem/system><PolicyNr>_<GoalNr>_<KPINr>

"Subsystem/System" can be "WH" for Ware House, "RM" for Remote Monitoring, "YM" for Yard Management or "VSP" for Virtual Service Platform (respectively Integration). PolicyNr refers to the unique identifier of the policy to which the goal refers to, GoalNr

5.3.1 KPIs for the Warehouse Subsystem

We here describe the KPIs and the metrics that were defined for the high level goal of the Warehouse Subsystem, which is to provide efficient and reliable storage and retrieval to its users. After a refinement step we defined 5 sub-goals with corresponding KPIs:

- G_WH3_1: Keep rejection rates of storage requests low (below 1%).
    - KPI_WH3_1_1: In this case the performance indicator is the rejection rate. To calculate the rate we need to measure the amount of requests that are made and the amount of rejections that are received in a given time frame of 1 day.
- G_WH3_2: Keep available storage capacity above 5%
    - KPI_WH3_2_1: In this case the performance indicator is the storage capacity. To calculate the capacity we need to collect events that signal the addition or retrieval of storage from the warehouse. These

events must contain the amount of storage effectively added or removed so that the total capacity can be kept up to date.

- G_WH3_3: Ensure that storage rejections are temporary. The system should avoid repeated rejection.
    - KPI_WH3_3_1: In this case the performance indicator is the number of times we assist to repeated rejections after five minutes. To calculate this we need to correlate 'reject' events with subsequent 'request' events that are made by the same user five minutes later. If the second 'request' event also generates a correlateable 'rejection' event we will have an indication that the sub-goal is not met.
- G_WH4_1: Retrieval should always be processed en-bloc
    - KPI_WH4_1_1: Every time there is a successful retrieval, the system generates an event containing a list of the items that were received, together with their quantity. This list needs to be compared with the list of items that were requested. In this case the performance indicator is calculated as the number of times we assist to retrieval events that list less storage than what was originally requested we find that the two lists do not match, plus the times we do not receive a retrieval event, given a request, within a configurable time frame (e.g., 10 minutes).. To compute this we need to capture and correlate retrieval request events, with a list of the storage being requested, with retrieval response events. If the response lists less storage we have an indication that the sub goal is not met. In particular we want to count how many of these situations occur in 1 day.
- G_WH5_1: Keep retrieval failures low
    - KPI_WH5_1_1: In this case the performance indicator is the retrieval failure rate. To compute this we need to collect the amount of retrieval requests made to the system in a given time frame of 1 day, as well as the amount of failures in that same time frame. Once we have both we can calculate the number of failures over the total amount of requests.

5.3.2 KPIs for the Remote Maintenance Subsystem

We here describe the KPIs and the metrics that were defined for the Remote Maintenance Subsystem's high-level goal, which is to provide efficient and reliable monitoring and reactions to emergency. After a refinement step we defined 5 sub-goals with corresponding KPIs:

- G_RM1_1: Monitoring should be done in near real time
    - KPI_RM1_1_1: In the use-case human operators that look at various monitors receiving live feed perform the monitoring. This sub goal is therefore partially solved through appropriate design. However the use case also states that the system will help the human operators by

changing the resolutions on the screens to guide them where they should look. An example is when a truck is leaving the yard the human operator should look at it. Therefore, the performance indicator is the amount of time it takes the system to modify the resolutions of all the monitors. To compute this we need to collect and correlate the event of a truck leaving the yard with the events of the resolution having been changed on all the monitors. We will aggregate them and calculate the average time it took over a given time frame of 1 hour.

  - KPI_RM1_1_2: There is also a second type of monitoring that is achieved automatically by aggregating events that signal a successful completion of one of the system's sub-processes. For example, we need to correlate when a truck enters the yard, with when it drops off some goods, and with when it leaves the yard. These three events need to occur inside a 15 minute window. In this case the performance indicator is the amount of time it takes the system to actually send out a success or failure event. Other similar situations should also be monitored and measured.

- G_RM2_1: In case of emergency the staff should be notified
  - KPI_RM2_1_1: This sub goal is also achieved partially by design since it may depend on the humans performing the monitoring, and sending out the notification. In this case we only measure the average amount of time it takes, the system to actually deliver the message.
  - KPI_RM2_1_2: There are also cases in which the system performs monitoring autonomously. In this case the performance indicator is the presence of an event indicating that the notification has been sent and received. In this case we want to count, given a time frame of 1 day, how many times the system failed to send an emergency notification within a given amount of time (e.g. 1 minute). We are also interested in capturing the average amount of time it takes the system to actually deliver this emergency notification.
- G_RM3_1: In case of accidents fire alarm should be notified
  - KPI_RM3_1_1: This policy requires that fire sensors be scattered around the environment. This must be given by design. To establish that a notification has been correctly achieved we need to be able to capture the sensors signalling that there is a fire. When we see that a sensor is signalling a fire, we want to witness an event from the alarm service stating that the notification was been received. The performance indicator is the amount of time that passes between the two events, which should never be more that 1 second.
- G_RM4_1: Corrective actions should be taken when an emergency is notified
  - KPI_RM4_1_1: In this case the performance indicator is the amount of times in a given time frame of 1 day that the human operators fail to take action on an emergency notification within 1 minute. A second

indicator is the average amount of time it takes a human operator to notify that he is taking action, after he receives an emergency notification.

5.3.3 KPIs for the Yard Management Subsystem

We here describe the KPIs and the metrics that were defined for the Yard Management Subsystem's high-level goal, which is to provide efficient and reliable truck management in the yard. After a refinement step we defined 8 sub-goals with corresponding KPIs:

- G_YM1_1: The Subsystem should maximize throughput of goods
    - KPI_YM1_1_1: In this case the performance indicator is the goods throughput, which is the rate of goods that are dropped off by the trucks in a given time frame of 1 day. To compute this we need to aggregate the events of successful delivery over one day.
- G_YM3_1: The distribution of notifications and the monitoring of the state of yard entities should be done nearly in real-time.
    - KPI_YM3_1_1: In this case the performance indicator is the average time it takes for notifications to be delivered to the subsystem's management. We consider trucks entering the yard, trucks obtaining or dropping off their loads, and trucks leaving the yard to be interesting notifications. To compute the average delivery time we need to capture and correlate the events indicating that the notifications were sent, and the events indicating that the notifications have been received.
- G_YM4_1: A smooth loading or unloading process should be guaranteed through advanced shipping notices.
    - KPI_YM4_1_1: In this case the performance indicator is the number of times we see correlated events for all the steps in a delivery or pickup. In this case we need to collect and correlate the events regarding the trucks entering the yard, the trucks picking up or dropping off their loads, and the trucks leaving the yard. In particular we want to count the amount of times that we are not able to correlate all these events in the given time frame of 1 day.
- G_YM5_1: The Yard Management Subsystem should know the position of all Yard Jockeys.
    - KPI_YM5_1_1: In this case the performance indicator is the number of times trucks fail to send a periodical update on their whereabouts. In the system we assume the trucks send an update with their location every 10 seconds, from the moment they enter the yard to the moment the leave it.

5.3.4 KPIs for Integration

This section describes the KPIs and the metrics that were defined for the Integration's high-level goal, which is to provide efficient and reliable integration of the three subsystems. After a refinement step we defined 6 sub-goals with corresponding KPIs:

- G_VSP3_1: The remote maintenance video stream is prioritized when the warehouse system or the yard management system is in error state
    - KPI_VSP3_1_1: In this case the performance indicator is the amount of time it takes the system to modify the resolutions of all the monitors. To compute this we need to collect and correlate the event signalling the error state with the events of the resolution having been changed on all the monitors. We will aggregate them and calculate the average time it took over a given time frame of 1 day.
- G_VSP4_1: The video stream of the yard reception is prioritized while a truck is doing reception process (Yard management <-> Remote maintenance)
    - KPI_VSP4_1_1: This case has already been treated in the subsection regarding the Remote Maintenance subsystem's KPIs.
- G_VSP5_1: The warehouse management gives goods storage process a higher priority, when there are too many delivering trucks on the yard.
    - KPI_VSP5_1_1: In this case the performance indicator is the presence of an event signalling that higher priority has been given to storage within 1 minute from the moment in which monitoring signals that there are too many delivering trucks in the yard. A second indicator is the average amount of time it took the priority to be switched after the monitoring signalled it should be. This is calculated over a given time frame of 1 day.
- G_VSP6_1: The warehouse management gives the goods retrieval process a higher priority, when there are too many empty trucks on the yard.
    - KPI_VSP6_1_1a: In this case the performance indicator is the presence of an event signalling that higher priority has been given to retrieval within 1 minute from the moment in which monitoring signals that there are too many empty trucks in the yard. A second indicator is the average amount of time it took the priority to be switched after the monitoring signalled it should be. This is calculated over a given time frame of 1 day.

5.3.5 INDENICA KPI and Rules Improvement Cycle

Policies, KPIs and Rules are subject to changes due to many different reasons:

- The business environment is changing
- The organisation's goals are changing

- The platform portfolio is changing
- Technology is changing
- The platforms are changing

All such changes affect the set of KPIs and thus also the monitoring and adaptation rules. But there are also changes to KPIs and rules that are intrinsic to the measurement system:

- Basic data for KPIs is not – or not sufficiently – available
- Analysis of KPI does not show the expected insight into the platform performance
- Monitoring data cannot be acquired as planned
- Monitoring results show unexpected behaviour
- Adaptation results do not show improvement of platform behaviour
- …

In all these cases the KPIs and rules have to be adapted, or dropped and replaced by new ones. In order to perform a systematic approach, we suggest a KPI and Rules Improvement Cycle as shown in Figure 26.

The Open Group SGF defines a SOA Governance Vitality Method that gives the frame for organisational transition to SOA. The INDENICA KPIs and Rules Improvement Cycle shall be a concretisation and a method for KPI and rules improvement.

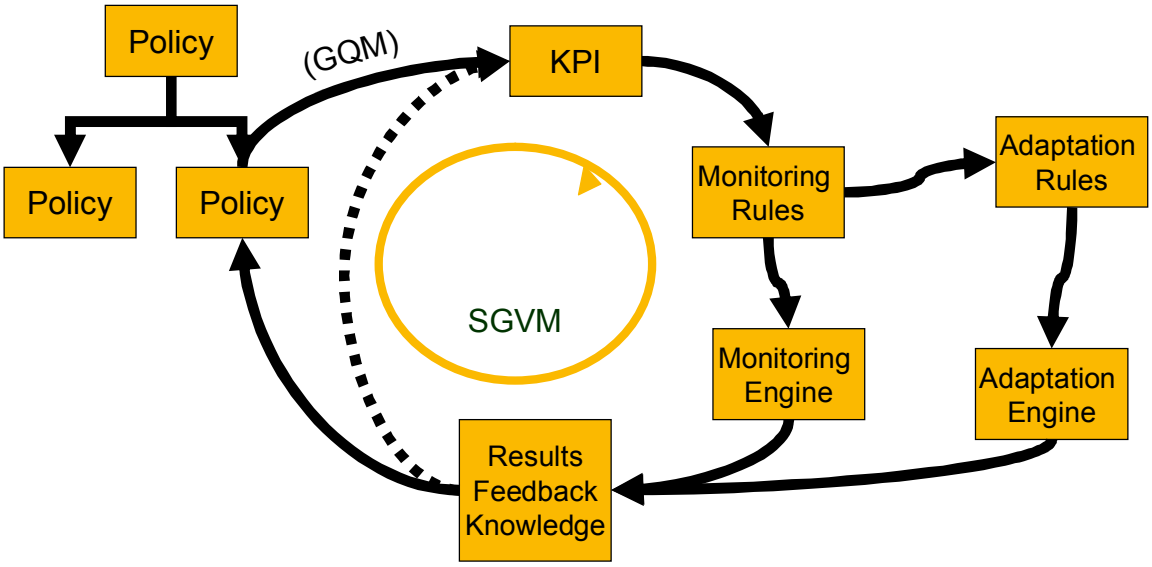

Figure 26: INDENICA KPI and Rules Improvement Cycle

In coincidence with the SGVM phases Plan – Define – Implement – Monitor we see the following tasks and involved roles in such an improvement cycle:

Phase	Plan KPI and Rules improvement
Goal	Define and adapt goals for monitoring and adaptation
Input	Policies and policy changes Regular platform KPI report

Tasks	Decompose policy and derive goals Analyse monitoring and adaptation results Adapt goals according to results
Output	Monitoring and adaptation goals
Role(s)	SOA Centre of Excellence (R)

Phase	Define KPI and Rules
Goal	Define set of KPIs and rules
Input	Monitoring and adaptation goals
Tasks	Apply GQM approach: Formulate questions to narrow down goals Identify measures and indicators
Output	(Updated) set of KPIs and rules
Role(s)	SOA Centre of Excellence (R) Platform Architect (A) Platform Integrator (A)

Phase	Implement KPIs and Rules
Goal	KPIs and rules ready for monitoring deployed platforms
Input	(Updated) set of KPIs and rules
Tasks	Translate monitoring and adaptation rules into formal language Deploy monitoring and adaptation rules on platform(s)
Output	Active monitoring and adaptation system
Role(s)	Platform Integrator (R)

Phase	Monitor KPIs and Rules
Goal	Insight into platform performance
Input	Active monitoring and adaptation system
Tasks	Collect monitoring data Collect adaptation incident data Compile data and report KPIs
Output	Regular platform KPI report
Role(s)	Platform Admin (R)

5.4 Monitoring INDENICA Governance

An important aspect of SOA governance is monitoring of services during runtime. This is essential for two reasons:

1) Monitoring the runtime environment provides information about the state of all running services. Even the basic fact of whether a service is running or

stopped provides valuable feedback to the Platform Administrator who can then verify if the running system is compliant with its design.

2) Monitoring can also include the performance of running services, by analysing runtime information events coming from underlying service platforms. This is essential in order to ensure that the services optimise their KPIs, which in turn helps prevent violations of Service Level Agreements or policies that the services are supposed to adhere to.

The direct consequence of a having a comprehensive monitoring system in place is the availability of information that allows for manual or automated adaptations to the underlying service platform. For example, the Platform Administrator can restart a stopped process, or an automated adaptation system can change the runtime configuration of a running service.

In INDENICA the monitoring process is more complex than in a standard service execution environment, as it needs to cover multiple service platforms and their interactions. However, the required end result is the same – the Platform Administrator should be notified of situations that violate polices or KPIs. INDENICA handles this task through the monitoring and adaptation framework, which does the following:

a) It aggregates monitoring information from all underlying service platforms.

b) It analyses collected monitoring information to detect or predict violations of KPIs.

c) It triggers automated adaptation actions to prevent or remedy detected problems.

d) Only in cases when automated adaptation is not successful the platform administrator is required to perform manual adaptation actions.

This design reduces the complexity of runtime administration of VSPs by introducing a significant degree of automation.

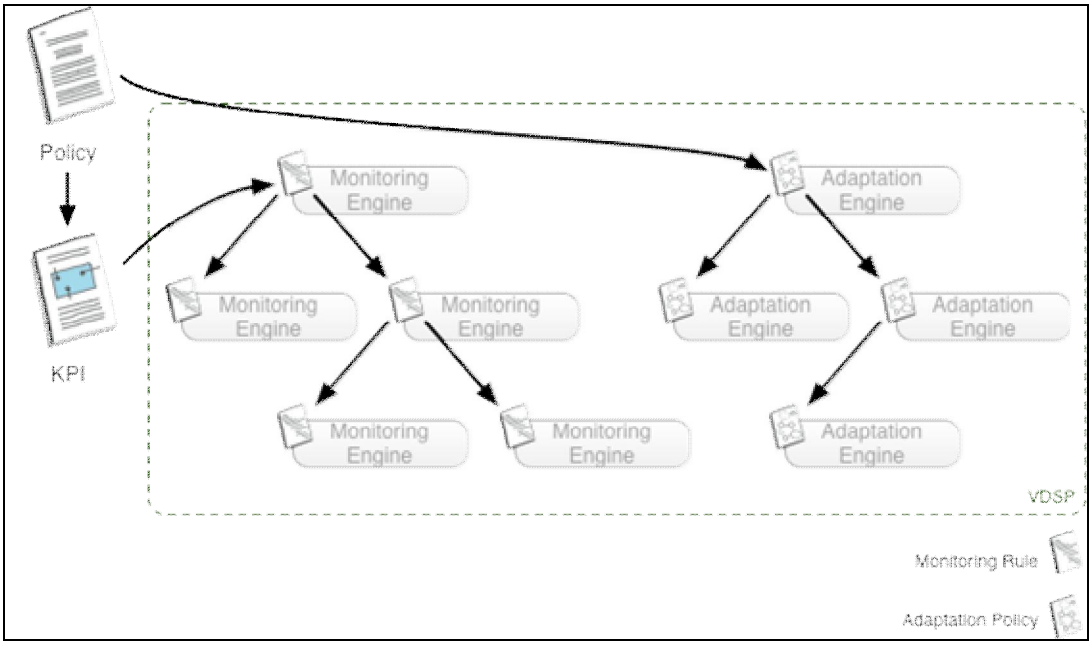

Figure 27: Policy and KPI Transformation Overview

At design time, the KPIs are transformed into high-level monitoring goals, used by the monitoring engine (as seen in Figure 27). A human operator, assisted by INDENICA tools (which introduce a certain degree of automation) can perform this transformation. The monitoring infrastructure employs a layered approach to achieve high-level goals by allowing for the arbitrary decomposition into fine-grained monitoring rules. This approach enables distinct rules to remain succinct and efficient, focusing on single service platforms or specific interactions. The rules are applied to an according hierarchy of monitoring engines (ME). Using this structure, low-level MEs that deal with details of service platforms or individual interactions, report their results to higher level MEs that are able to perform their task using aggregated data, without the need for information about platform specifics.

Furthermore, Figure 27 illustrates that Governance policies are used in a similar fashion within the INDENICA adaptation infrastructure. Governance policies define the high-level adaptation goals that can be arbitrarily decomposed to allow for fine-grained control of the VSP. Analogously to the monitoring infrastructure, the adaptation policies are applied to an according hierarchy of adaptation engines (AE). Low-level AEs are concerned with service platform and/or infrastructure specifics, whereas high-level AEs can manage the VSP using abstract goal specifications.

Monitoring rules can be associated with appropriate adaptation decisions (i.e., a particular adaptation action, or notification of the system administrator). Moreover, *Adaptation Actions* are annotated with an estimate of what costs each action would have on the performance of the overall system, to allow the system to make automated decisions.

Formal Descriptions of Policies and KPIs

The task of converting KPIs into *Monitoring Rules* and *Adaptation Actions* is not trivial and has to be performed during design time by appropriate experts. One of the reasons for this is that KPIs are represented as text based descriptions using

natural language, and as such cannot be processed automatically. However, in the setting of INDENICA tools will be provided to ease the process of translating KPIs into *Monitoring Rules* and *Adaptation Actions*, by introducing a formalized description for the specification of KPIs.

6 Conclusion and Outlook (SIE)

In this INDENICA deliverable we have analysed the landscape of governance types and drafted a better structure of a governance ecosystem. Based on this ecosystem we analysed the requirements for governance in the context of IDENICA which has to tackle the challenges of product line engineering approaches and of service orientation at the same time.

The overall structure of such governance is already well described in the Open Group SOA Governance Framework. For the specific context of IDENICA we specified a number of additional and more detailed role description, processes and key performance indicators.

In further work of the INDENICA project these detailed specifications can be of use for conducting the evaluation of the case studies. Especially the KPIs can be introduced into the monitoring system, described in formal languages and thus be a means to give evidence of the benefits of introducing a VSP.

There are also aspects that this work did not yet cover but for which it could be the basis for further research and standardisation. TOGAF is a framework for architectures; the referenced SOA Governance Framework of The Open Group is by its title focused on Service Oriented Architectures; Khusidman evaluated its suitability for BPM Governance. This plethora of governances and frameworks with numerous overlaps and contradictions is still confusing and a more clear structure and basic definitions would be of great benefit. A first attempt was done jointly by The Open Group, OASIS and OMG and published in [Kreger, Estefan 2009]

Going down from the meta level to details, all governance descriptions in literature state the importance of being based on best practice, but the level of detail is often too coarse to be really relevant for daily work. Thus a best practice framework for roles will be of benefit especially when it comes to continuous improvement of governing and governed processes.

Table of Figures

References

[Agile 2012] The Agile Manifesto, see http://agilemanifesto.org/ accessed 19.01.2012

[COBIT 4.1] COBIT®, Control Objectives for Information and Related Technology, edited by ISACA, http://www.isaca.org/Knowledge-Center/COBIT/Pages/Overview.aspx

[Deming 1986] Deming, W. Edwards: Out of the Crisis, MIT Centre for Advanced Engineering Study, 1986.

[INDENICA D1.1] Report on State of the Art in Service Platform Design, Adaptation, Deployment and Monitoring, INDENICA Deliverable D1.1, http://www.indenica.eu/fileadmin/INDENICA/user_upload/d11-stateoftheart.pdf

[INDENICA D1.2.1] Requirements Engineering Framework, Language and Tools for Service Platforms, INDENICA Deliverable D1.2.1, http://www.indenica.eu/fileadmin/INDENICA/user_upload/d121-reqengfrw.pdf

[INDENICA D3.1] View-based Design Time and Runtime Architecture for Tailoring VSPs, INDENICA Deliverable D3.1, http://www.indenica.eu/fileadmin/INDENICA/user_upload/d31-viewbasedarch.pdf

[INDENICA D5.1] Description of Feasible Case Studies, INDENICA Deliverable D5.1, http://www.indenica.eu/fileadmin/INDENICA/user_upload/d51-casestud.pdf

[Khusidman 2010] Vitaly Khusidman: BPM Governance Framework; BPTrends, June 2010; http://www.bptrends.com/publicationfiles/ONE%202010-ART-BPM%20Governance%20Framework-VKhusidman-v51.pdf

[Kreger, Estefan 2009] Heather Kreger, Jeff Estefan: Navigating the SOA Open Standards Landscape Around Architecture; White Paper by The Open Group; June 2009; https://www2.opengroup.org/ogsys/jsp/publications/PublicationDetails.jsp?publicationid=12196

[OECD 2004] Organisation for Economic Co-Operation and Development: OECD Principles of Corporate Governance; downloaded 19.10.2012 from http://www.oecd.org/dataoecd/32/18/31557724.pdf

[Ontology 2009] The Open Group: The Open Group SOA Ontology; Technical Standard. June 2009; www.opengroup.org/projects/soa-ontology

[SCRUM 2012] see http://en.wikipedia.org/wiki/Scrum_(development) accessed 19.01.2012

[SEI 2009] Soumya Simanta, Ed Morris, Grace A. Lewis, Sriram Balasubramaniam, Dennis B. Smith: A Scenario-Based Technique for Developing SOA Technical Governance; Technical Note CMU/SEI-2009-TN-009, June 2009; http://www.sei.cmu.edu/library/abstracts/reports/09tn009.cfm?DCSext.abstractsource=SearchResults

[SGF 2009] The Open Group: The Open Group SOA Governance Framework; Draft Technical Standard, 2009. www.opengroup.org/projects/soa-governance.

[Stal 2011] Stal, Michael: Architecture Governance; In: Hitchhiker's Guide to Software Architecture and Everything Else, 2011, http://stal.blogspot.com/2011/10/architecture-governance.html

[TOGAF 2011] The Open Group: The Open Group Architecture Framework (TOGAF); Technical Standard V9.1, 2011; http://pubs.opengroup.org/architecture/togaf9-doc/arch/

[Wikipedia 1] See http://en.wikipedia.org/wiki/Governance, accessed 19.01.2012

[Wikipedia 2] See http://en.wikipedia.org/wiki/SOA_Governance accessed 19.01.2012

[Wikipedia 3] See http://en.wikipedia.org/wiki/Responsibility_assignment_matrix accessed 19.01.2012

Appendix: Guideline for a SCRUM process in Safety Critical Development Environment

A) Role Landscape

In the safety critical development environment special knowledge concerning regulations and independence are necessary to complete the actual work, release an increment to the customer and to certify compliance with the regulations.

According to ISO 13485 there must be an appointed *management responsible who is responsible for established QMS processes* and awareness of regulatory requirements throughout the organization.

Teams usually have difficulties with fulfilling those conditions within the pure SCRUM rules and face a challenge.

In order to comply with these requirements, introduction of following roles is recommended:

Role	Responsibility / Required Skills
Risk/Hazard Manager	Responsible for Risk analysis, risk measures, risk evaluation, complaint handling
Quality Manager (QM)	Responsible for supervising regulatory environment of the project, validation plan, for competent authorities; the QM can be the Scrum Master, if appropriately trained. In any case he/she needs to be independent from the development.
Software Architect	Responsible for system structure, architecture/design requirements, interfaces
Product Owner (PO)	As defined by SCRUM

Since these 4 roles need to have very special knowledge and have dedicated safety related responsibilities they are considered essential; other roles could be covered by the team as well.

According to the SCRUM rules, *the team is responsible for delivering the product*, and is made up of 5–9 people with cross-functional skills who perform the iteration work: analysis, design, development, test, technical communication, documentation, etc. Teams shall be self-organizing and self-led, moderated by the SCRUM Master.

A number of skills are required within the team for interaction and collaboration with the four supervisory roles:

- Coach for the Risk/Hazard Manager
- Tester(s)
- Tester responsible for all testing documentation (especially for system testing)
- Developer

- Configuration Manager responsible for the build process, operation concept
- In case there are more SCRUM teams to coordinate (SCRUM of SCRUMS), install a coordinating role that take care of the SCRUM teams: Coordination Manager
- Coach for documentation and regulatory support,
- Tracker for Requirements Tracking, Hazard Tracking
- Tracker for Software Problems and Resolution
  Background Note: IEC 62304 requires a 'Software problem resolution process' that ensures the creation of a 'problem report' for each problem detected in a software product. It is recommended to clarify with the quality management department how this normative requirement is implemented in the organization's QMS.
- Responsible person for project and progress tracking
  Background Note: A Project Management Plan / Quality Management Plan and person responsible for them are required by the regulations.

B) Role Allocation

Use the first iteration to clarify role allocations and personal responsibilities, especially of Risk Manager, Quality Manager, Software Architect, Product Owner, Test responsible, and Configuration Manager.

If all roles that are required by regulations for reviews are allocated to different individuals in the SCRUM team, then reviews, releases etc. can be handled easier:

- through the iteration review meeting (incl. minutes),
- through the (documented) check of the DONE criteria,
- or through the iteration planning meeting (as review for the requirements)

C) Project Planning and Control

The main overall document for project control and project planning is the project's Quality Management Plan. It shall be released before starting agile methodology.

When writing the Project Quality Plan, it should be clear how and following to which process the project will start and which process is going to be applied in the respective project. In the course of the project, the Project Quality Plan can be updated according the respective Change Management rules, if necessary.

Be precise on the definition of the length of iterations and stick to the defined length (since you have to follow your own regulations).

Adapt numbers of iterations in the plans (Quality Management Plan), if necessary.

Have regular validation points (regarding process, product …)!

D) Archiving the Project Quality Plan

Archive the first Product Backlog and the first Iteration Backlog (for planning and requirements reasons as well). Archive the Product Backlog and the Iteration Backlog at the beginning of each iteration, since they are the basis for iteration planning.

Then the iteration planning can be done and updated at the start of each iteration. There can be different layers of planning (day, iteration, release, product, portfolio planning).

Have a complete released Product Backlog of the implemented Product Backlog items at the end of the last iteration. Document process tailoring in the QMP.

E) Requirements Management

Product Requirements are collected continuously in the agile Product Backlog. For each iteration a part of the prioritized Product Backlog is selected, detailed and worked on. It is possible that not the whole Product Backlog is implemented in the finally released product.

Have a Product Backlog released which was valid at the start of the first iteration (which is also necessary for Project Control).

Have a complete Product Backlog of the implemented Product Backlog items released at the end of the last iteration.

For the requirement specifications it is possible to use, e.g., the product backlog with the features, stories, sub-stories and exactly described, specific & detailed use cases, and exactly described test cases which have to match together, according to prior agreement with the responsible quality management department and in compliance with the applicable processes.

Even if a product backlog contains all the listed information and therewith all necessary design input information required by regulations, the local applicable processes have to be followed (including the templates and forms required by the applicable QMS (quality management system)).

F) Traceability

There must be a Requirements tracing report according to IEC/ISO 62304. In agile context as well, the following traceability requirements have to be fulfilled:

Traceability between system requirements, SW requirements, SW system test, and risk control measures implemented in SW shall be addressed in the SW development plan. Especially, traceability of Risk Keys / Safety Keys is of importance down to Code, Test specification and Test record.

Example in the agile context: If user stories are broken down step by step, then the reference to the super-ordinate requirements must be maintained.

G) Release Management and Tests

Release Planning and Release Backlog are essential for the Release Management. Release management must define acceptance criteria, version management, and configuration management. Release Management must be documented.

Test phases required by regulations are: acceptance tests, functional tests, design-level tests, test cases.

For this purpose define

- How will the SW be verified: unit tests, module tests, integration tests, system tests, test evaluations, consequences?
- Which test specifications and which test performances are executed when (including documentation)?
- Which tests run in a certain iteration, which tests run in every iteration, which tests run in the non-agile part of the development process (if there is so)?

Define the release criteria for product release. This is usually defined in the QMS off the “agile world part”. Please note that theses release criteria are not the acceptance criteria for the user stories, but product release criteria like e.g. criteria regarding quality, test coverage.

Fix version management and configuration management. This is usually defined in the QMS off the “agile world part”.

Distinguish in agile development between official (complete, valid) releases and in-official iteration results. Official releases must have all respective review & release documentation.

Definition of DONE: when the tests passed, etc.

It is possible to define DONE differently for official and in-official iterations results, but make sure that not too much workload is deferred to ‘some later’ iteration. Of course, in any case DONE contains e.g. that test are passed and that the software is deployable.

Perform Formal, documented Testing after Code Freeze. Hints for test execution:

- For regression tests automated tests suites are important for each release. Try to get as much test automation as possible.
- Continuous integration is recommended.
- Perform code inspections and reviews on the most critical code (risk-based approach).

H) Configuration Management / Documentation Management

Configuration Management and Documentation Management follow the organization’s overall configuration and documentation management process.

To reduce documentation effort in an agile process with its many short iterative working cycles, it is recommended to establish an automated documentation management system with effective tool support.

Hints:

- The Configuration and documentation management system provides a validated archiving system.
- Configuration management system can be applied for source code and documentation.
- The Configuration/documentation management system should provide digital signature.

J) Review-Release-Process and Document control

Purpose of a Review is to reduce the risk of errors through a check performed by independent and competent individuals.

Since in the agile process most specifications are changed in each iteration, a full review documentation of the changed documents in each iteration would create a lot of effort.

In order to keep the benefit of a review, one way would be, to fully review the first version when the specification is created and to fully review the very last version of the specification. During the iterations the changed specifications are reviewed by competent persons to find mistakes and are corrected correspondingly. During the iterations the documentation of the review is delimitated to the pure information (what was reviewed, who reviewed, what was found …).

How this review process is done, is described as part of the document control procedure, document management procedure or similar. This process description defines which documents have to be created and have to have which state (draft, reviewed, released …) in which process step (end of requirements definition, start of implementation, end of the agile process part …). Besides, the/a procedure defines the change proceeding of documents.

Write “iteration review minutes” which could document the review of the documentation created during the respective iteration.

Use digital signatures: One group creates a backlog (e.g., requirements), and this group and an additional competent (regarding quality, independency and subject competence) individual signs this backlog version (status) electronically in the respective Configuration Management System.

K) Process Descriptions, Procedures and Work instructions

Write simple clear clean procedures, with clear purpose and principles. Write them as detailed as necessary and as rough as possible:

- A software development process includes SW development planning, requirements analysis, architectural and detailed design, implementation, verification, integration, integration and system testing, and software release. [For details see IEC 62304]

- They should define the activities *in a degree* that the persons who need to apply it, are able to understand, apply, follow and live it so that the regulatory requirements are fulfilled, e.g., through defining the most relevant process steps, activities with inputs and outputs, roles and responsibilities involved, documentation produced.
- Clarify with the respective Quality Department the applicable QMS structure and how generic the process description should be and how detailed it should be to serve as applicable process description for use in practice.
- Fix a quite detailed (but "live-able") process description (especially for 'agile' methods) (incl. templates, methods, responsibilities, process steps, necessary reviews, milestones…), in order to ensure that the regulatory requirements are met and to facilitate clear and consistent answers in potential audit situations.
- Determine detailed and exactly the deliverables for 'agile' methods (e.g.: Are Iteration acceptance minutes/ Iteration review minutes written?).
- Determine, how special documentation requirements are handled (e.g.,: Traceability matrix, Risk analysis table, risk control measures, RMP, Design review, code reviews, unit tests, regression tests, software requirement specifications, software architecture specification, software design specifications, software integration testing specification, software system testing specification)
- Define, document, implement, and use a rigorous product risk management process (which is required by ISO 14971, ISO 62304, and ISO 13485). Systematically apply management policies, procedures, and practices to the tasks of risk management, such as analyzing, evaluating, and controlling risks. Document the essential activities in a risk management file and the SW safety classification in classes A, B, and C.
- Preconceive tailoring project-specific process possibilities or project specific details of common procedures, etc. Tailoring Examples:
- Number of 'release iterations' or number of iterations between the 'release iterations' can be different depending on short-lasting or long-lasting projects.
- Insert additional / delete milestones for project quality control in very long- / short-lasting projects.
- Plan dedicated iterations to be "bug-fix-iterations" with no new feature development.
- Define templates rules for meeting minutes (e.g. for including photographs)
- Where possible allow room for improvement for the agile teams in the process description.
- It is not enforced to define one single everlasting version of a process in the QMS and assign it at project start and never change or improve it.

- Processes shall be improved and changed, e.g. through retrospectives or reviews, but it must be ensured that the process changes stay within the tailoring rules of the initial version of the process
- In case of more significant process changes, the new process description shall be released in the QMS including transition rules. For significant changes, close cooperation with the Quality Department is indispensable. In general, there shall be a change process in the QMS that has to be applied for process changes as well.
- Discuss the process descriptions regularly in the agile retrospectives and adjust them to 'theoretical' requirements and practice and adapt the descriptions.
- Train people in details that have to be executed day-to-day.
- Be aware that there are documentation requirements by the organization's Quality Management System procedures and there are product-related documentation requirements due to product liability and market approval / clearance.