Business as a service multi-layer governance architecture

Juan Li

N° d'ordre 2014ISAL0027
Année 2014

Thèse

Business as a Service Multi-layer Governance Architecture

Présentée devant
L'institut national des sciences appliquées de Lyon

Pour obtenir
Le grade de docteur

Formation doctorale
Informatique

École doctorale
École Doctorale Informatique et Mathématiques

Par
Juan LI
(Ingénieur en Informatique)

Soutenue le 17 mars 2014 devant la Commission d'examen

Jury MM.
Prof. Olivier PERRIN (Rapporteur)
Prof. Bruno VALLESPIR (Rapporteur)
Prof. Dominique RIEU
Dr. Selmin NURCAN
Prof. Youssef AMGHAR
Prof. Frédérique BIENNIER (Directeur de thèse)

Laboratoire de recherche
Laboratoire d'InfoRmatique en Image et Systèmes d'information (LIRIS)

Business as a Service Multi-layer Governance Architecture

Résumé

Pour faire face aux enjeux d'une économie mondialisée, aux fluctuations du marché et aux changements de la demande (personnalisation massive, qualité…), les entreprises recourent de plus en plus aux stratégies de collaboration et d'organisation en réseau et adoptent des stratégies orientées « produit/service ». Cette tendance est renforcée par le développement des applications du Web 2.0 (voire 3.0?) et l'adoption d'architectures orientées services permettant d'augmenter l'interopérabilité et l'agilité des systèmes d'information. En outre, les possibilités offertes par le Cloud Computing permet de rendre le déploiement plus flexible. En parallèle, le développement de stratégies industrielles comme le « lean manufacturing » et le 6-Sigmas permet d'améliorer les procédés, l'organisation industrielle elle-même et la qualité des produits.

L'objectif de ce travail de recherche est de coupler la vision « industrielle » à la vision « système d'information » traditionnelle pour permettre de mettre en place un modèle de services industriels composables, orchestrables et « gouvernables ». Pour cela, nous proposons de mettre en place une architecxture de gouvernance globale « connectant » les différentes couches du système (métier/industriel, service, plateforme et infrastructure), permettant d'améliorer la gouvernance du système globale (en évitant les incohérences liées à une prise en compte et une optimisation « isolée » des différents facteurs de performance) tant au niveau organisationnel que technologique. Ceci pourrait permettre d'améliorer les performances tant au niveau « métier » que « technologique », augmenter l'agilité du système et supporter plus efficacement les stratégies de collaboration en développant une approche basée sur la sélection / composition / orchestration de services métier industriels.

Mots-Clés: Service-Oriented Architecture – Cloud Computing – Business as a Service – Gouvernance en Multi-Cloud – Non Fonctionnelle Propriété – S ervice Oriented Organisation Industrielle - Système de Produit Service – Lean Thinking - Qualité de service - Accord de Niveau de Service

Business as a Service Multi-layer Governance Architecture

Abstract

Due to the renewed globalised economical environment and the market evolution (mass customization, sustainability requirements…) the call for developing product-service strategy becomes a major stake, leading industrial companies to set collaborative business organizations and develop business services. This trend has been favored by the large-scale IT environment provided by the web 2.0 and by the development of interoperable and rather agile IT technologies based on services leading to SOA-based information systems reorganization. At the same time, lean and six sigma theories have also been used in industries to improve the industrial process itself so that profitability, quality and reputation are increased. As a new economical and technical model, Cloud Computing has generated a tremendous amount of interest and excitement in recent years as it gives a new and useful way to address IT challenges

To achieve the primary goals of these technologies, concepts and models, an efficient industrial organization governance method is necessary. We propose a flexible, efficient, low cost monitoring strategy, it can couple the different layers of economic ecosystem (including business strategies, business/industrial/IT services, execution platforms and infrastructure means) it can overcome existing industrial governance architectures' limits (most of them are rather "fixed" and lack agility, overall perspective governance as they have unilateral perspective), and it could drive the industry towards better practices, improve ability of enterprises to cope with changes from both a technical and an organizational point of view, as well as reinforce external and internal collaborative work of enterprises.

Key-Words: Service-Oriented Architecture – Cloud Computing – Business as a Service – Governance in Multi-Cloud – Non Functional Property – Service Oriented Industrial Organization – Product Service System – Lean Thinking – Quality of Service – Service Level Agreement

Acknowledgements

Without the support of many people this dissertation would not have been possible. Therefore, I would like to acknowledge their support throughout three and a half years.

First of all, I would also like to thank Professor Olivier PERRIN and Professor Bruno VALLESPIR for their review, Professor Dominique RIEU, Mrs Selmin NURCAN and Professor Youssef AMGHAR to have accepted to evaluate my work. I thank them for their time and the suggestions to improve this dissertation.

I especially thank my doctoral advisor Prof. Frédérique Biennier, without her guidance this dissertation cannot be achieved. I would also like to thank her for offering such a great working environment, introducing me into the international scientific community, and for supporting both my publication efforts and my doctoral dissertation. She always had time and so patient for me when I needed her views on specific topics. She supported and gave me valuable advices not only on my research working but also on my scientific life and keeping positive attitude.

I would also like to thank all the professors and my collegues in my team, I do appreciate the cooperation expriences I had with them. Especially, my thanks go to Professor. Youssef Amghar, Professor Chirine Ghedira, Dr. Wendpanga Francis Ouedraogo, it was my honor to achieve co-authered articles with them.

Furthermore, I gratefully acknowledge the financial support from Chinese Scholarship Council (CSC) to grant me the scholarship for three and a half years study.

Finally, I owe my deepest gratitude to my family. My parents always supported me throughout my life and I am much obliged to them for this.

Table of Content

List of Acronyms

Terminology	
1.	**Defined Name**
2.	**Defined Components**
3.	**Defined Requirements**
4.	**Defined Resources**
5.	**Defined Patterns**
6.	**Defined Policy Rules**

1. Defined Name		
1)	BaaS	Business as a Service (we extended XaaS to business as a service level)
2)	BDM	Business Decision Maker(BDMs raise BaaS management and governance requirements and make business decisions)
3)	CSF	Critical Success Factor
4)	Multi-layers of our governance architecture	
	BDL	Business Decision Layer
	BPL	Business Process Layer
	BSL	Business Service Layer
	BIL	Business Infrastructure Layer
	GEL	Governance Execution Layer
5)	FP	Functional Property
6)	IMGB	Integrated Management Governance Bus (Proposed communication and collaboration middleware)
7)	MLA	Multi-level Agreement (we extended Service Level Agreement to Multi-level Agreement)

		BPLA	Business Service Level Agreement (the lower level agreement of our extended multi-level Agreement)
		BSLA	Business Process Level Agreement (the upper level agreement of our extended multi-level Agreement)
8)	NFP		Non-functional Property
9)	NFR		Non-functional Requirement
10)	Proposed Management Framework includes two models		
		BROM	Business Resource Organization Model (select convenient resources to build business scenario)
		NGPM	Negotiation and Governance Preparation Model (negotiate to achieve multi-level agreements)
11)	QR		Quality Range (define thresholds to distinguish and manage different performance situations, it aims to reduce violations of agreements)
		SR	Satisfied Range (performance is satisfied in this range)
		TR	Tolerable Range (performance is tolerable in this range)
		AR	Alert Range (performance is unexpected in this range)
		TH_{S-T}	Threshold between satisfied range and tolerable range
		TH_{T-A}	Threshold between tolerance range and alert range
12)	QoS		Quality of Service
13)	R		Resource (in our architecture, resource includes BP, Task, service, operation and infrastructure)
14)	SP		Service Provider
15)	SO-MGA		Service-Oriented Management and Governance Architecture (our multi-layer management and governance architecture)
16)	XaaS		Everything as a Service

2. Defined Components

1)	AE	Action Engine (AE is a component of our governance architecture, it is designed to correct and modify resources' performance and to improve their quality)
2)	KPI	Key Performance Indicator (it is defined in our monitoring model, KPIs are used for implementing generated monitoring policy rules to monitoring quality of resources' execution)
3)	Aggregator	Aggregate Key Performance Indicator's initial monitoring results to comprehensive results
3. Defined Requirements		
1)	BS-req	Business Service Requirement (formalized BS-reqs are used for selecting convenient services)
2)	Ser-req	Service requirement (in our deconstruction process, a Task-req should be deconstructed into series of Service requirements)
3)	Task-req	Task requirement (in our deconstruction process, a BP requirement should be deconstructed to series of Task requirements)
4)	MonReq	Monitoring Requirement (it is defined in our Monitoring model)
5)	CompReq	Computing requirement (it is defined in our computing model. Computing requirements request compose Key Performance Indicator's monitoring results to comprehensive reports)
4. Defined Resources		
1)	BP	Business Process (it is a type of resource which defined in our multi-level agreement)
2)	Task	It is a type de resource which is defined in our mutli-level agreements
3)	Inf	Infrastructure (it is a type of resource which defined in our multi-level agreements)
4)	OP	Operation (it is a type of resource which defined in our multi-level agreements)

5)	Service	We define service as a type of business resource, it is defined in our multi-level agreements
5. Defined Patterns		
1)	Pat	Transform Pattern (it is defined in our monitoring model, it is used for transforming monitoring requirements to governance policy rules)
2)	GPat	Governance pattern (it is defined in our monitoring model. Governance patterns are invoked by generated monitoring policy rules, they deploy convenient Key Performance Indicators to execute monitoring policy rules)
3)	CompPat	Computing pattern (it is defined in our computing model, CompPats are used for transforming computing requirements to computing rules)
6. Defined Policy Rules		
1)	PolR	Monitoring policy rule (it is defined in our monitoring model; all the monitoring rules are generated from received monitoring requirement)
2)	CompRule	Computing rule (it is defined in our computing model, CompRules are used for implementing received computing requirements)

Part 1 Introduction and State-Of-The-Art

1 General introduction

1.1 Motivation and Challenges

In today's highly collaborative business context, the global competition has been forcing organizations to strengthen their core competitiveness. Business model has being evolving from a pure product perspective towards an integrated product-service orientation [Cavalieri et al. 2012]. This trend makes providing products alone is insufficient in terms of remaining competitive for firms developing. Servitisation allows to add competitive value to product. As Product Service Systems may integrate product and support services, it leads to improve satisfaction of customers [Beuren et al. 2013]. To be efficient, such Product Service Systems require internal and external cooperation. This involves that organizations have to produce customized integrated business solutions which can increase business profitability, reduce the consumption of process, and then ultimately enhance market share and comprehensive competitiveness. To fit such a dynamic context, agile manufacturing has been concerned as a new paradigm for successful manufacturing enterprises since 1990s. [Kidd, 1994] defines agile manufacturing as "an organizational ability to thrive in a competitive environment characterized by continuous and sometimes unforeseen change". Since then, nine major categories are defined to contribute the development of agile manufacuturing including Information systems, supply chain and other collaborative organisations, product and manufacturing systems design, business practices and processes, facilities design and location. Due to their organizational flexibility and adaptation abilities, collaborative organisations play an important part in such vision. This requires a flexible dynamic and integrated mechanism to manage information flow to support efficiently the frequent and dynamic interactions among partners. As a consequence, this impacts the entreprise Information Sysrem organization to accommondate reconfigurability and composability to integrated information flow between partners, to support specific business needs (such as production, design, ordering, etc), to establish automated architecture, to facilitate information exchange and communication technology [Luis M, 2001]. IT has also to address new business challenges in a proactive way to facilitate the application of services for integrated business solution. As a consequence IT is considered as a critical success factor to add value and makes an enterprise's products competitive [Weill and Ross 2004]. With decades' development, agility is still attracting an increasing amount of attention in industrial areas. A number of forces can be stated which are driving the evolution of agile manufacturing in

business. Such as the increasing competition, fragmentation of mass market, cooperative business relationships, changes in customer's expectations, the increasing societal pressures [KEKE, 2012].

Focusing on customer requirements also involves to pay more attention on the production value chain, avoiding useless activities (i.e. any activity that does not add value to the product / service delivered to the client), leading to lean organisations. Whereas increasing continuously the production quality may lead to Six-Sigmas adoption. Both of these organisations alsoi require an efficient and reconfigurable Information System to support the industrial production process organization. As a consequence the IS agility directly impacts the agility of business organization. From a review of agile manufacturing research [Luis M, 2001], we can see that since the beginning of agile manufacture research, the Information Systems attracted the most attention, it was attributed the largest number of citations.

This increasing IS importance while building high quality and agile industrial organization, not only leads organizations to focus on product value chain, but also involves paying more attention on information value chain and adopting Lean six sigma thinking to information system organization. Fitting the industrial agility while keeping IS profitability involves that the agility of information value chain is a critical factor. The alignment of IT and business can mutually promote ISs' organization and business performance. To this end, one can associate industrial resources to their IS artifacts. This leads to select, compose and orchestrate these IT artifacts to fit the industrial ressources oraganisation.

Service-Oriented Architecture (SOA) and Cloud Computing provide new solutions to narrow the gap between IT and business, and to organize business resources. However, these loosely coupled paradigms also bring uncertainties in industrial organizations. How may we guarantee that enterprises can gain expected benefits (required values can be increased) from their applications without unexpected side effects (wastes, defects and violations); and how can we make sure all associations could be maintained and operated in a right way that is high-efficiency, reliable, auditable, customized and trustworthy?

To fit this challenge, our research objective is to propose a Service-Oriented industrial organization which aims at improving the agility of information systems for product-service systems and then ultimately improving the agility of business (including agility of information value chain and agility of business process). In addition, this service-oriented industrial organization also aims at governing and improving the quality of loosely coupled business resources (from business to IT) and the entire business value chain.

To achieve this primary objective, there are some related works that should be discussed. Since a lot of similarities exist between Lean and Agile manufacturing, the requirement of combing Lean, Agile and Sustainbale paradigms has been proposed. When the aspects of Lean, Agile, and Sustainable Manufacturing are considered as one system, Lean system emphasizes the system stability, minimizes waste, requies a flexible production and a continuous improvement strategy whereas Agile system should pay attention on capability to unpredictable changes and increasing collaboration. [Mason-Jones, 2000] proposes the "Leagility" concept wich aims at maximizing profits by combining the advantages of lean and agile manufacturing. This combined concept brings ideas to extend the meaning of agility in traditional agile manufacturing, it drives the propagation of agile manufacturing concept to build agile enterprises, and eventually to realize an agile industry fitting noval business paradigms [Koho, 2010].

Extending the Agile System to IS organization requires defining precisely agility. This may be complex as there is no consensus yet. A variety of views on business agility provide some common aspects with their own differences. [Canter 2000] compares agility with flexibility and suggestes that unlike flexibility which is responsiveness to anticipated contingencies, agility is (in business vernacular) the ability of organization to thrive in competitive marketplaces by continuous, accelerated and often unpredictable changes. [Conboy and Fitzgerald 2004] compares agility with flexibility and leanness. They define flexibility as the continual readiness of an entity to proactively or reactively embrace changes and leanness as the maximization of simplicity, quality and economy which requires eliminating all waste. In their work they state that agility only requires waste to be eliminated where its ability to response to change is not hindered. As such they conclude that agility is a combination of flexibility, speed and leanness. [Van Oosterhout et al. 2007] defines business agility as the ability to sense highly uncertain external and internal changes and to respond to them reactively or proactively. This relies on innovation of the internal operational processes, involving the customer in exploration and exploitation activities, while leveraging the capabilities of partners in the business network.

According to these definitions, in our research we take 'Agility' as a combination of flexibility, speed, quality, leanness, customization and self-improvement by making improvement in any performance objectives.

Improving and ensuring this promised agility of collaborative organisations and its support information systems and business processes motivate our research goal that aims at proposing a new Service-Oriented Industrial Organization which can manage, govern and improve the quality and agility of industrial / business resources as well as their IT artifacts. According

to the lean vision, we focus on the value chain organization and governance. To achieve this goal, we model the value chain as a process used to transform input resources into "products" (real tangible products or services), fitting the customer requirements. As a consequence, the associated IT artifact is defined as the IT support process where each component implements the IT support tasks associated to the industrial / business component involved in the value chain. So to govern the target value chain, we propose to monitor and govern the associated IT components chain depending on the business objectives to provide a clear understanding of business and IS quality and performance. To this end, we need to monitor the system and define some performance measurements. Enterprise performance measurement interests both the customer and competitors. Different kinds of performance indicators can be considered and associated to the different flows (financial ones, delay, quality…). Moreover, as lean and agile manufacturing systems often lead entreprises to focus on their core business and develop collaborative strategies, the global performance must also be monitored. Paying attention to these collaborative organisations, performance measurement and metrics pertaining to Supply Chain Management has received increasing attention from researchers or practitioners. Using the literature and results form this field, [Gunasekaran, 2004] developed a framework to promote a better understanding of the importance of SCM performance measurement and metrics. [Berrah, 2012] proposes a systemic approach combined with the SCOR model to define a performance evaluation approach. Each sub-system is described in terms of its constituting business processes, can be analyzed. Both works show the importance of coupling process models to performance measures while organizing the global governance / decision support system. To fit this goal, ECOGRAI is a method based on GRAI models which aims at designing and implementing Performance Indicator Systems for industrial organizations. It allows to identify relevant Performance Indicator (PI). Thus, it reduces the number of PI with improving efficiency and performance measurement system adoption. [Doumeingts, 1995].

Extending this (collaborative) industrial/business governance background, to loosely coupled IT and business resources governance in integrated product-service industry, requires taking into account cross-organizational cooperation, complex collaborative relationships between business process and services as well as new challenges due to the IT implementation context, namely SOA/Cloud environment, such as control of out-of-house infrastructure elements that current IT governance approaches do not fit [Niemann et al. 2009]. Beyond IT governance, Service-Oriented Governance should also pay more attention on the impact of IT quality on business quality and the impact of collaborative participants' quality on the quality of enterprise's entire value chain.

However, current works of service oriented governance or governance in cloud do not fully cover challenges related to product-service requirements. For example [Papazoglou, 2006] provides a formal model to express visibility constraints, manage compliance and configure cloud resources on demand but this model does not allow a dynamic reconfiguration at runtime depending on the context which is an important requirement in a dynamic collaborative organization. Focusing on the IT side, different governance means and methods are developed. While optimizing the "real" resource utilization at runtime, a particular attention must be paid on the way "elastic QoS" is defined in order to avoid penalties due to the risk of deviation from the agreed QoS level [Jayasinghe, et al., 2012].

This increases the call for a global governance system allowing an efficient execution and support of collaborative business processes. This involves monitoring both infrastructure and services, as well as monitoring both information system and business performance, taking into account SLAs, elasticity, QoS, etc. [Clayman, et al., 2010]. As trust, managerial capability and technical capability have significant relationships with cloud-deployment performance [Garrison, et al., 2012], SLAs should be understood by both cloud expert and non- expert so that common performance indicators can be recognized and associated to these different non functional properties. Unfortunately, as stated in the SLA survey made by [Alhamad, et al., 2011], SLA frameworks are focused on technical performance and do neither take into account security nor other business related non-functional properties. Moreover, resources measuring techniques need further refinement to be applied in the cloud context in order

1) to ensure some level of trust between service providers and customers,
2) to provide a flexible and agile way to tune performance metrics parameters,
3) to support real costs evaluation means.

To this end, [Katsaros, et al., 2012] proposes a self-adaptive hierarchical monitoring mechanism for clouds. This framework implemented at the Platform as a Service layer, allows monitoring the QoS parameters based on SLA for business benefits. Nevertheless it lacks of providing a flexible policy enforcement mechanism and it does not indicate its scalability in web service framework. While considering quality from a "customer" point of view, [Jureta, 2009] proposes a comprehensive quality model for service-oriented systems. It allows specifying the quality level, determining the dependency value and ranking the quality priority. However, the performance issues related to cloud resources are not discussed and details are missing regarding the correlation of the quality model with the service cost model. Lastly, the monitoring

mechanisms should be non-intrusive and should let a minimum runtime footprint over the execution environment [Heward, 2010]. It has also (1) to fit the cloud elasticity to keep up when an application or infrastructure scales up or down dynamically [Gogouvitis, et al., 2012] and (2) to provide a "simple" interface so that the cloud complexity is as transparent as possible for end users. This requires being able to "generate and deploy on the fly" contextual monitoring means.

1.2 Research Issues and Contribution

Our background analysis shows the importance of extending the agile industrial requirements to the associated IT support to be able to answer to the ever-changing market requirements efficiently. This also requires to monitor and to govern the industrial/business system and its IT artifact. Overcoming these existing limits and fitting these challenges require a Service-Oriented Governance Approach to build flexible, dynamic, customized, and cloud-ready monitoring processes for both Information system agility and business agility. This raises the general question of this thesis:

How do we achieve this required Service-Oriented Governance Approach with dynamic, customized and cloud-ready monitoring mechanisms?

To answer this general question, we propose a multi-layer governance architecture supporting our business integrated governance loop (see Figure 1).

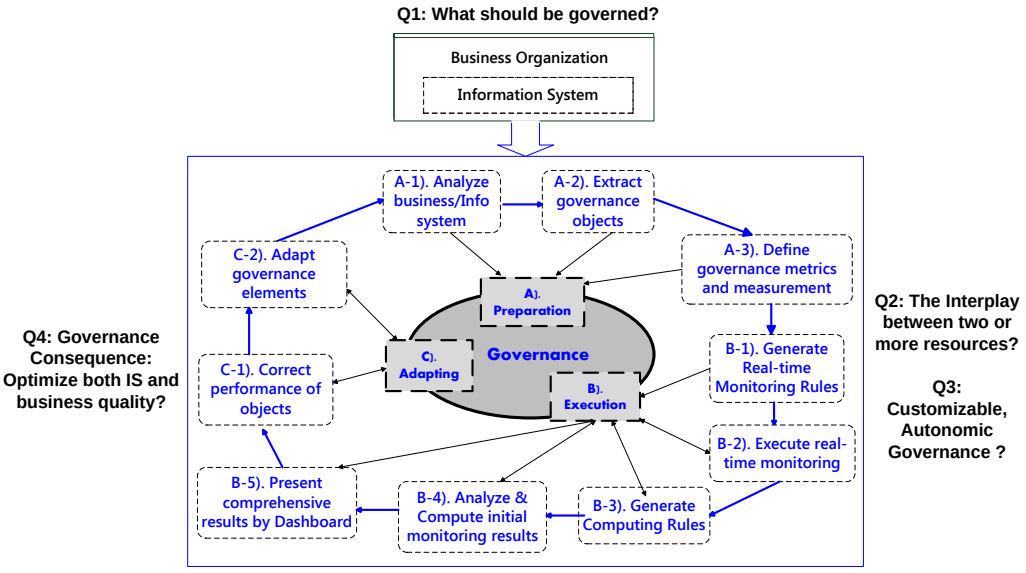


Figure 1 *Overview of Our Proposed Governance Loop*

This governance loop includes three high-level phases: A) Preparation; B) Execution and C) Adapting. To resolve this general research question more precisely, we divide these three phases into more detailed ones:

- Preparation phase includes 3 detailed phases: 1) Analyze model of systems, 2) Extract governance objects, 3) Define governance metrics and measurement.

- Execution phase includes 3 detailed phases: 4) Execute real-time monitoring, 5) Analyze & Compute initial monitoring results, 6) Present comprehensive results using dashboard.

- Adapting phase includes 2 detailed phases: 7) Correct the performance of governance objects, 8) Adapt governance elements to fit new requirements.

Analyzing the general research question and building a governance approach, lead to clearly understand the monitoring objects. Different monitoring objects drive the different monitoring mechanisms, thus to design an appropriate and high-efficiency monitoring mechanism we have to identify monitoring objects accurately. Consequently **we have first to identify what should be governed, i.e. to define and select the IT ressources associated to real business/ industrial resource used in the value chain.**

To answer this sub-question, we have to understand what is the most importance factor that impacts business performance. In the current business landscape, value chain is a key element of the lean strategy for enhancing the enterprise's performance, delivering superior total value to the customer in terms of speed, cost, quality and flexibility [Ketchen, 2008]. Agile value chain performs better business outcomes. With the development of SOA and Cloud Computing, the Information System has been taken as the backbone of an enterprise, thus besides traditional product value chain, the information value chain becomes the key driver for entire enterprise's performance. Therefore, the value chain, especially the information value chain is the most concerned monitoring object for this required Service-Oriented Governance Approach.

Porter [Porter 2008] described that a value chain is a chain of activities that a firm operating in a specific industry performs in order to deliver a valuable product or service for the market. In current business environment, an organization can have one to few value chains. A value chain is usually de-composed into several business processes. These business processes can be divided into several sub-processes. Depending on the nature of the sub-process, it may contain sub-sub-processes or even lower levels' processes. Therefore, to monitor enterprise's value chain, we have to monitor the associated enterprise's business processes. Based on the definition of manufacturing performance indicators that can be used to monitor the integrated product-service value chain, we need a method to define and select convenient Key Performance Indicators to pick some measures associated to the IT artifact/ business resources. Besides,

specifying measurement, to evaluate integrated product-service value chain, we also need a method to compose related Key Performance Indicators to provide synthetic measurement results according business requirement. As far as the "cloud consumer" vision is concerned, a business process is usually decomposed into one or more business tasks. A business task usually requires one to a few services and a service is implemented by operating infra-structures. For this reason, monitoring business process requires monitoring all related business tasks, services and infrastructure elements as they are essential objects.

To govern these objects **we have to understand the way governance is organized and how these objects' performance impact on the performance of enterprise's value chain.**

These governed objects are organized and deployed in business processes depending on their Functional Properties (FPs). Understanding the organization of these objects requires a multi-layer model to arrange these objects into value chain fitting the SOA layered model. Functional interdependencies of these objects express the process of enterprise's value flow. Besides the Functional Properties' organization, managing value chain's quality requires paying more attention on these governed objects' Non Functional Properties (NFPs) which constrain Functional Properties achievement and deal with quality of these governed objects. These different NFPs (that are related to the different industrial performance indicators) are measured thanks to Key Performance Indicators associated to required business resources and their associated IT artifacts. However, the organization of Functional Properties impacts the management of Non Functional Properties. Therefore, the organization of these objects' Functional Properties should be taking into account to design governance of these objects' Non Functional Properties. Moreover, measuring Non Functional Properties precisely requires an efficient, quantifiable and trackable Non Functional Property management approach. However Non Functional Property's management is facing open issues, such as lack of appropriate solutions to refine Non Functional Properties within inter-related Functional Properties. To overcome these challenges, we propose a Non Functional Property classification which aims at classifying Non Functional Property into detailed Critical Success Factors (CSFs). After refining Non Functional Property to evaluate the value chain performance level, we design composition algorithms which take into account the Non Functional Properties and functional interdependencies to compose the relevant Non Functional Properties' monitoring results to comprehensive and meaningful results for customers.

As Non Functional Properties have different priorities in different fields of industry, the Non Functional Properties management and governance should be customizable. This leads to the next research question: **how can we make the**

governance approach be customized and self-adjusted to meet different requirements? To answer this question, we propose a governance requirement formalization and governance policy rule generation process, taking advantage both of Model Driven Engineering and of Pattern-based Engineering. Formal models are associated to requirements and used to generate contextualized policies that will be used at runtime to select and orchestrate the governance elements accordingly. A self-management strategy allows the governance process to orchestrate governance elements automatically in the dynamic environment. Furthermore, a customizable presentation provides clear view of both real-time and periodic results from operational level to enterprise level. Thanks to a customizable mashup dashboard.

Of course, providing the governance results allow Business Decision Maker (BDM) to understand business status and to be able to make business decision efficiently afterwards, but **how governance approach can improve the quality of Business Process automatically and to avoid unexpected results before they occur?**

Resolving this problem requires a non-invasive and autonomic management. As we have discussed previously, we need to pay attention on the essential governed objects including business tasks, services and infrastructure elements. In SOA and Cloud environment, enterprises are collaborating with a large number of partners. Business processes require diverse services. Therefore the performance of service's selection and management impacts the organization's value chain and business outcomes. An automatic business resource selection, re-usage and management strategy is required to simplify business management and to optimize value chain by orchestrating the most appropriate resources.

Due to the large scale of collaboration, the increasing number of Service Level Agreements (SLAs) makes the agreement management become an onerous work, whereas the violations of SLAs impact business outcomes. This requires an efficient agreement management approach which can manage and mediate a large number of SLAs. This management should have a negotiation and transparent reconfiguration strategy to reduce violations and the number of reconfiguration on enterprise side, as well as agreement should cover Non Functional Property management and QoS. To this end, we extend traditional SLA to multi-level agreements which includes Business Process Level Agreement (BPLA) and Business Service Level Agreement (BSLA) associated to different quality ranges to narrow the gap between business and technology, as well as, to reduce the violations of agreements.

Besides building multi-level agreements, in order to achieve automatic governance, we propose an immunity inspired strategy to manage governance elements automatically and to react on unexpected situation proactively.

To sum up, the main objective of this PhD work is to set an industrial service oriented organization governance loop to support large scale networked and collaborative strategies overcoming technological and organizational limits, focusing on networked value creation to fit the business requirement dynamically.

To achieve this objective, we propose a dynamic Multi-layer Service Oriented Management and Governance Architecture (SO-MGA). This architecture includes:

1) a Business as a Service Management and Governance Preparation Framework which includes a) a Business as a Service Resource Organization Model and b) a Business as a Service Negotiation and Governance Preparation Model;

2) a Business as a Service Governance Execution and Adapting Framework which includes a) a Multi-layer Monitoring Model and b) a Multi-layer Computing Model.

This architecture is designed thanks to five sub-contributions.

Sub-Contribution 1: To solve the sub question **Q1** (what should be governed and how to organize these governed objects), we consider and extend the 3 traditional layers of cloud (Software as a Service; Platform as a Service; Infrastructure as a Service, etc.) with a "Business as a Service (BaaS)" level to meet business governance requirements. Managing BaaS, we pay attention on quality of value chain, and build a multi-layer value chain management model to manage governed objects in the value chain. Due to the feature of product service industry, the governed objects include business processes, business task, services, operations and infrastructure elements. All these governed objects are defined according to their characteristics. They are deployed into the different layers of our model to express the organization of value chain.

Sub-Contribution 2: To solve the sub question **Q2** (how these objects' performance impact on the performance of enterprise's value chain?) we extend the traditional Service Level Agreement (SLA) to Multi-Level Agreements (MLAs) which include Business Process Level Agreement (BPLA) and Business Service Level Agreement (BSLA). We design a MLAs negotiation model. This negotiation model integrates the inter-related Functional Properties of cross-layer's governed objects. These dependency relationships show the interaction between these resources and the impact of individual object's quality on the quality of entire value chain.

Sub-Contribution 3: After building the multi-layer governed objects management model and the multi-level agreements negotiation model, sub question **Q3 (**how can we make the governance approach be customized and self-adjusted to meet different requirements?) is solved thanks to the definition of the Governance as a Service paradigm, using a Non Functional Property

governance policy rule generation and execution support. We define a governance requirement formalization strategy and a generic policy rule generation process, coupling Model Driven Engineering and Pattern-based engineering strategy to transform customized governance requirements to specific governance policy rules. These generated governance policy rules are used to orchestrate governance elements to manage governed objects' Non Functional Properties and to govern the quality of these governed objects. At runtime, the generated monitoring policy rules containing all the Non Functional Property governance requirements, they are used to orchestrate Governance Execution Elements. Computing rules containing all composition requirements are used for computing initial runtime monitoring results to integrated meaningful governance results.

Sub-Contribution 4: To resolve the sub question **Q4** (how governance approach can improve the quality of Business Process automatically and to avoid unexpected results before they occur?) we design an automatic correction strategy to adjust performance of governed objects accordingly. This correction strategy thanks to Action Engines that act on governed objects according to the defined multi-level agreements. Furthermore, to achieve agility and self-management ability, we take advantage of autonomic management to strengthen the ability of governance autonomic management, self-adaptability, self-organization, as well as self-learning. We adapt artificial immunity theory to define and control governance elements' lifecycle and quantity.

Sub-Contribution 5: Lastly, to complete the proposed Multi-layer Management and Governance Architecture we have to support flexibility and non-intrusive transparency of the governance process. To this end, we define an Integrated Management and Governance Bus (IMGB) with a unified data format. This IMGB allows message exchange and collaboration between governance components to achieve the governance framework flexibility. It also supports to "plug in" and "pull off" dynamically the components of our governance architecture.

1.3 Thesis Outline

This dissertation consists in two parts: Part one introduces General Introduction and State of the Art and Part two presents our contribution.

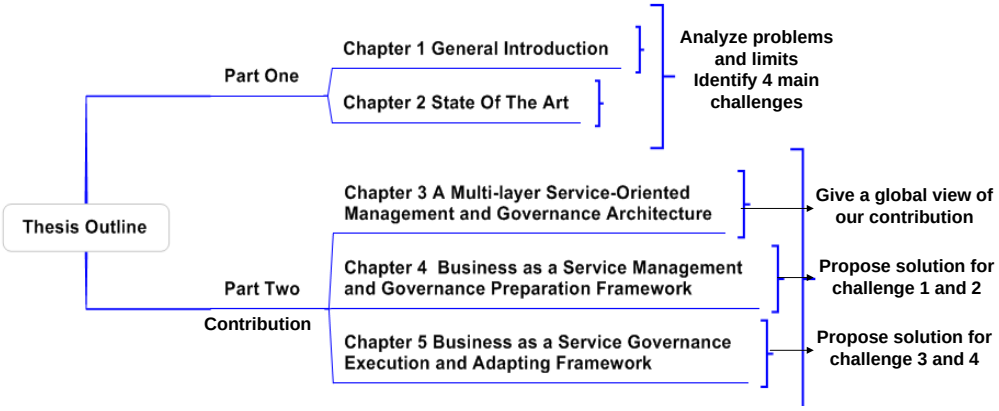


Figure 2 *Thesis Outline*

Chapter 2 (State of The Art) identifies existing limits according to our challenges. It is organized in different sections:

- Section 2.1 introduces the global context, putting the stress on the role of information system (IS) and information technology in today's organization. ISs agility is the key driver for business agility. ISs agility approach is inspired by the successfully practice of Lean Six Sigma in manufacture. However there are particular challenges that need to be addressed to integrate ISs agility and business goals.
- Section 2.2 presents the way business process management can be coupled with enterprise architecture. Business agility requires the people, processes, strategy and technology of an organization to be able to respond dynamically to changes, paying attention on the way business processes are managed and how involved resources and orchestrated. Business Process Management (BPM) is a key point for organizations managing all resources and activities during business processes execution. To address system complexity and narrow the gap between business and IT, Enterprise Architecture (EA) may be used to provide a platform for BPM to orchestrate organization's resources accordingly and to implement organizational level strategies. Combining BPM and EA for better identifying business value chain is an efficient way to achieve business agility. However, there are still some limits such as EAs do not provide a clear solution to govern the established enterprise architecture whether match the global business goals or not. BPM lacks the architectural principles, policies, and standards that emerge and develop during the link to strategy and business model and then through the architecture lifecycle..
- Section 2.3 introduces the Service Oriented Architecture. To achieve the alignment between business and IT, SOA as an architecture style is well suited for modern EA. It is essential to align business and IT agility. Due to the flexibility provided by the selection/composition, orchestration mechanisms and the inter-operability provided by defacto standards for services, SOA can

support collaborative IS organization and management for large scale organizations, and reduce costs while organizing cooperations. SOA governance and performance measurement is the essential requirement, but there are still some challenges need to be addressed, such as changing management to customer perspective and ensuring quality of service.

- Section 2.4 introduces agility implementation thanks to Cloud Computing. Cloud computing can be seen as a natural evolution of the widespread adoption of virtualization, SOA, autonomic, and utility computing. Everything as a Service (XaaS or EaaS) is a critical concept for cloud computing to implement its key enabling technologies (fast wide-area network; powerful, inexpensive server and high-performance virtualization). Cloud environment requires a cloud-ready performance measurement to assist organizations govern their business performance in complex and large scale environment.
- Chapter 2.5 discusses traditional performance measurements. They are not dynamic enough, cannot fit the cross-layer, cross-domain requirements in the emerging Cloud environment. A Cloud-ready performance measurement is required and it needs to be autonomy, scalability, adaptability. To achieve an autonomic governance solution we analyze existing autonomic management methods and the way immunity inspired methods can fit our multi-layer governance challenge.

2) Part two details our Multi-layer Service-Oriented Management and Governance Architecture (SO-MGA). Our contribution is continuously illustrated with a Logistic Use Case.

Chapter 3 presents the global view of our multi-layer management and governance architecture. It consists of Business as a Service (BaaS) management framework and BaaS governance framework. We enrich the traditional XaaS with Business as a Service (BaaS) layer and visualize all business resources to organization's business processes. To facilitate governance and simplify the interaction between governance components, we design an Integrated Management and Governance Bus and the general definition of governance objects. To specify the governance, we also introduce the Non-functional Properties classification.

Chapter 4 introduces the proposed Multi-layer BaaS management framework. It consists of two models: Business Resource Organization Model which manages and organizes business resources, and BaaS management negotiation model which mediates and manages Multi-level Agreements. The Logistic Use Case is used to demonstrate the implementation of these two models.

Chapter 5 presents the proposed Multi-layer BaaS governance framework which includes Multi-layer Monitoring Model and Multi-layer Computing Model. We detail our customizable monitoring model which includes

requirements formalization, automatic governance rule generation, and autonomic management for governance elements' lifecycle and algorithms for governance composition. The consistent Logistic Use Case is used to demonstrate the implementation of our monitoring and computing models.

The last chapter of this thesis (chapter 6) contains work summary and an outlook of future works.

2 State Of The Art

2.1 Global Context

2.1.1 Information System Agility is the Key for Business Agility

The ability to improve business performance is one of the most critical driving forces for organizations. This ability has been introduced as 21st Century Manufacturing Enterprise Strategy [Nagel and Dove, 1991]. Improving business performance, enterprises continually strive to increase production. In recent years, the effects of this effort have demonstrated that providing products alone is insufficient in terms of remaining competitive [Beuren, et al. 2013]. The Product-Service System (PSS) represents a competitive opportunity for many companies as they seek to reduce consumption by altering how their products are used by providing services. [Cavalieri et al. 2012] presents several claimed benefits associated with integrated product-service business solution:

- An increase of revenues, as services tend to have higher profit margins and can provide a stable and countercyclical source of revenues;
- A differentiating weapon for competing in mass-markets characterized by commoditized technologies and products,
- A decreasing variability and volatility of cash flows throughout the life of a product, allowing for a higher shareholder value.

However, although integrated business solutions are thought to deliver higher margins, most organizations find it quite problematic to master the integrated product-service business solution [Cavalieri et al. 2012]. The basic idea behind the PSS is that it drives the business focus from the design and sales of physical products to the design and sales of a system consisting of products, services, supporting networks and infrastructure elements, which are jointly capable to fulfill customized requirements. This may lead to new distributed and collaborative organisations as focusing on core business may increase the profitability whereas developing outsourcing / collaborative partnership is necessary to provide a global Product-Service solution fitting the customers' needs. Supply chain connects via vendor-customer relationships the ultimate customer to the ultimate supplier, as a "system of systems" [Berrah 2012]. As a consequence, supply chain performance drives the performance of global business organization. Besides, strengthening competitiveness for enterprise and enhancing organizational profitability require flexibility and reactivity to response to changes efficiently.

Since 1990s, the agile manufacturing paradigm was formulated in response to the constantly changing and as a basis for returing to global competitiveness. [Goldman, 1994] defines agility is an overall strategy focused on thriving in an unpredictable environment. As reported in the agile manufacturing research review provided by [Luis M, 2001], a large attention is paid on the associated information system to support efficiently such collaborative business organization. Nevertheless, by now there is by no consensus on what exactly business agility is nor how one could assess and achieve business agility. Some researches try to give a definition of business agility, such as [Tallon, 2008] that suggests that an agile business is one as "responsiveness to changes in demand, new product development, and change in product mix, product pricing, market expansion, supplier selection, IT adoption and diffusion". [Wadhwa and Rao 2003] describes the differences between agility and flexibility and how flexibility and agility overlap. They suggest that flexibility is focused on a single system and is defined as a predetermined response to a predictable change on the opposite. Whereas agility is focused on groups of systems and entails an innovative response to an unpredictable change. [van Oosterhout et al. 2007] suggests that "Business agility is the ability to sense highly uncertain external and internal changes and respond to them reactively or proactively, based on innovation of the Business Agility: Need, Readiness and Alignment with IT Strategies internal operational processes, involving the customer in exploration and exploitation activities, while leveraging the capabilities of partners in the business network." Even though a consensus on a definition of business agility has not yet emerged, taking all of existing definitions into account, we can see the concept of quickness and speed is the main concern of business agility. Speed is one of the critical success factors for business. It can be required in various areas, such as time to market for new products, time to process a customer request, time to participate to collaboration, time to reconfigure business process and time to recovery from fault, and so on.

This agility requirement constrains enterprises to build customizable support information systems that can be aligned on the dynamic industrial process organization to achieve competitive product-service solutions. Consequently improving the business performance not only requires improving production processes, but also requires improving the quality of services and the support information systems, integrating information flows between partners. This requires to adapt the information system so that it can be aligned with the supply chain organisationto improve the process execution. It has also to provide information about parameters that assess specific goals of particular business strategy [Qrunfleh, 2014].

With the development of IT, the capabilities of information system have become the key enabler for enterprises to achieve their quickness and speed, enabling enterprises to anticipate, adapt and respond to whatever complexity the global economy, competition and changing technologies present, and to develop the flexibility to embrace change and thrive in this marketplace where rules change every day.

To address these challenges, modeling plays a very important role to organize and analyse the complex collaborative business networks built to achieve a common goal. [Vernadat 2002] suggests that Enterprise Modeling (EM) is the art of externalizing enterprise knowledge which adds value to the enterprise or needs to be shared. Various EM methods like PERA, CIMOSA, GRAI or GERAM have been proposed to deal with enterprise reengineering, customer satisfaction, process management, integration and coordination (see Table 1).

Table 1 *Comparison of Enterprise Modelling Methods*

Method	**Goal**	**Views**	**Models organisation**
CIMOSA [CIMOSA, 1993] -Open System Architecture for CIM	building an integrated production system.	functional information organization	Multi-dimensional organization, paying attention to the derivation axis (from generic to particular models)
GRAI [Chen 1997] - Integrated Methodology	Building a decision system	Processus, decision, information and physical resources	Propose conceptual, organizational and implementation models
PERA [Williams, 1994] - Purdue Enterprise Reference Architecture	Building entreprise functional and organisational models including human resources to	Information oriented view (planificationn, scheduling and control) and operational views (related to the product/service production)	7 layers associated to the engineering steps from the requirements capture to the operational implementation.
GERAM [IFIP 1999] - Generic Enterprise	Building a consistent framework integrating	Modelling Methods and entrepise models depending on the	Generic and partial models, 7 steps engineering method

Reference Architecture and Methodology	other methods	different methods	

Most of these methods are designed to improve the production system organization, decreasing production time and improving collaboration between different aspects such as function, decision, information, etc [Goepp, 2008]. However, there are still several issues that are not fully addressed:

- Flexibility and dynamicity: these methods are designed to model entreprises according to a static perimeter, focusing on a “To be” vision. As a consequence, they cannot handle continous changes in enterprise processes and it is difficult to adjust their components to fit novel market requirement.

- Ensuring a global quality: these methods provide “prescriptive” models and do not integrate any performance measurement nor governance organisation. To overcome this limit, [Doumeingts, 1995] proposes to extend the GRAI method to design ECOGRAI. ECOGRAI is a method to design and to implement Performance Indicator Systems (PIS) for industrial organizations related to the models built with GRAI, but it does not specify the metrics and the implementation is complex and do not fit lossely-coupled rources organisations.

To overcome these limits, we need to consider the features of different components, unify their global goals and pay attention on the importance information systems. We can take advantage of the view from different levels (strategy level, tactic level, operational level) to specify monitoring and governance requirements. We also need to establish specifiy metrics and measurement to enrich these views with specific quality monitoring. Moreover, we have to define a clear connection of information value chain with real business resources. Since information systems become an important support system for business, the entreprise operational models should be related to information technology provided that the business resources are related to IT artifacts. This requires paying attention on IT and information value chain.

IT development increases the importance of IT support for the different corporate processes, from accounting to manufacturing or shipping activities, leading to set the IS as a key enabler to smooth and improve internal and external of enterprises. The ability of an enterprise to adapt its IT capacities to market changes is increasingly suggested as an important organizational capability [Sengupta and Masini 2008]. This means organizations must focus on building business capabilities to leverage IT to gain a competitive advantage. Information systems have been replacing traditional communication media for internal and external communications extending information value chains beyond traditional enterprise boundaries. An organization’s information

infrastructure is the key to business agility, Due to business executives cannot detect and respond effectively to change without access to the right information, at the right time, by the right people.

The quality of information value chain closely affects the quality of business processes. For example, [Bailey and Francis 2008] presentes a case study based research suggesting that information transparency and collaboration have positive effects on enterprises value chain performance. However, building a dynamic information flow management needs an inter-disciplinary approach, combining the technical and relational aspects from the respective fields of system dynamics and collaboration in order to deliver improved organizational business performance. [Choe 2008] suggests that information flow management has a positive impact on quality improvement, a high dependability of supply and cost reduction. [Tsai, 2013] studies on the performance impacts of information system technology on e-retail industry. This work suggests that the focus on value chain activities enables us to examine the complementarities between different parts of the value chain from a sourcing perspective in the e-Retail context. The fast-paced change in the technology and service offerings of various industries requires further understanding of the challenges to improve the value of information flow, to add information value to business value chain and to enhance business outcomes performance.

The business value is also dependent on the “support systems” and as a consequence relies on IT organization. Improving IT organization is a way to improve quality of information value flow, and then as a result business performance can be boosted. [Seethamraju and Sundar 2013] suggests that as a key component of current IT infrastructure in a majority of organizations today, ERPs have delivered cost efficiencies, control and consistent execution, support many changes that take place during firms’ business processes. [Raschke 2010] suggests that IS derives value to the enterprise, the high quality of information value can achieve business process agility. As business and its supporting IS are aligned, understanding the component of business agility is important to select and organize resources of IS to implement IS agility. [Sambamurthy et al., 2003] suggests that business agility is composed of Operational agility, Partnership agility and Customer agility. [Sambamurthy et al., 2003] defines Operational agility as “the ability of firms’ business process to accomplish speed, accuracy and cost economy in the exploitation of opportunities for innovation and competitive action.” Partnering agility is the leveraging of collaboration relationships. [Raschke and David, 2005] defines Operational agility as the ability to add and /or reconfigure a business process by quickly adding new capabilities to the set of business process capabilities to accommodate the potential needs of the firm.

[Raschke, R., 2010] suggests that business process agility is composed of four components: re-configurability, responsiveness, employee adaptability and a process-centric view. Re-configurability is defined as "ability to deploy superior new configurations of functional competencies that better match the environment" [Pavlou and EI Sawy, 2006]. Responsiveness is the ability to identify and recover from change [Sharifi and Zhang 2001]. Employee adaptability is the ability for people to adapt to fit different situations. Process-centric view is the ability to consider the management perspective at business process level, have a better understanding of the entire process crossing different functional domains in an end to end vision [McCormack and Johnson, 2010].

In advanced economies, organizational performance depends on both the technological innovation and the organizational changes enabled by technological innovation [Brynjolfsson and Hitt 2003]. Quantitative empirical studies reported that certain industries attain higher IT productivity impact and greater cost reduction than others, providing further support for the inclusion of industry characteristics as an important contingent factor influencing the realization of IT business value [Melville et al., 2004] [Elbashir et al., 2008]. [Raschke, R 2010] suggests that IT infrastructure flexibility is positively related to business process agility. The performance of business process is closely linked to the capabilities, applications, and the management objectives for deploying IT and IT infrastructures. Business processes are the means to achieve the overall goals of the organization, so business process level performance is the essential factor to impact the organizational performance [Elbashir 2008]. Therefore, IT impacts the internal organization level performance, IT must be able to change, to have ability of decision-intelligence and able to blend IT requirements and capabilities fully into the business strategy and build agility into the enterprises.

In a summary, high-efficiency information infrastructure is an increasingly important element of business products and services and the foundation of enterprise wide processes [Weill and Ross, 2004]. Especially with the rising of e-commerce and e-business, Information System is definitely one of the critical factors for business market. Hence, it is important for auditors to assure that ISs are adequately controlled, secured, and functions as intended. To meet these requirements in the modern business environment, it is necessary to have a complete understanding of the way enterprises organize their IS and business activities. To improve the performance and efficiency of business processes, information infrastructure activities should be able to review frequently to ensure that system activities parallel business activities. However, enterprises are facing the increasing complexity and variety of IS along with serious challenges [Huang 2010]. High efficiency IT infrastructure consists of "the leadership and organizational structures and processes that ensure that the

organization's IT sustains and extends the organization's strategies and objectives" [W."RP" Raghupathi, 2007]. Information System governance, hence, is an instrument of strategic business-IT alignment [Hirschheim and Sabherwal, 2001]. Organizations with an instituted information governance process are more effective at seeking, collecting, processing and applying information and are getting more value from their and others' information sources [Kooper, 2011].

As we have discussed, the value of information flow is increasing the importance in organization's production system. Implementation of IT agility has a significant impact on the value creation process of business agility. IS agility is a critical success factor for business agility. An agile IS should allow business process to fit speed, accuracy and cost economy requirements, be able to reorganize individual components, to combine individual tasks to respond to environment changes. With the increasing internal and external collaboration requirement, IS allows enterprises to manage their business resources more efficiently. Besides the traditional physical product value chain, information value chain plays a more and more important role for business outcomes. The performance of information flow chain dominates the performance of entire enterprise value chain and impacts the organizational performance. After understanding IS agility leads to business agility, contemporary enterprises are making significant investments in IT (such as web services, data management, customer relationship management, supply chain management or enterprise resource management) to leverage the functionalities of these technologies in shaping their business strategies, customer relationships, and strengthen business competitive.

However, the technology, business environment and market requests are ever-changing. How enterprises can make their ISs are agile enough for fitting those changes and how can they make sure these investments enhance business performance as they want? We should pay attention on finding an approach to implement and monitor IS agility to meet enterprises' business requests.

A strategy to achieve IS agility consists in maximizing IT activities' value for business requests and minimizing non-value activities in information value chain. This strategy matches the core philosophy of lean thinking: Maximize Value, Minimize Waste. Lean thinking has been successfully used in various manufacturing, construction and other fields for decades. Besides using Lean thinking to manage the traditional physical product flow, Lean thinking has been more and more practiced in information flow to increase performance of business process by improving quality of information flow with the development of e-commercial and emerging business paradigms.

2.1.2 Lean Thinking for Information System

As we have discussed previously, even enterprises have noticed the importance of IS/IT for business, and we understand the agility of ISs is the key for business agility. Achieving ISs agility, the management of ISs is still facing particular challenges in the ever-increasing volume of information. With an efficient management, a big volume of valueless information can do nothing but increase cost, disturb business decisions, reduce business efficiency, and so on. Therefore, to build a successful IS, we have to continually optimize ISs organization and improve the performance of ISs. Lean think is a way to improve the performance of ISs, as a consequence it can also improve the organization of business, thus provide efficient and effective business processes. In this sub-section we introduce the application of Lean approach to specify information value, analyze and optimize ISs, the improve ISs management and the application of Six Sigma approach to pursue perfection of ISs. The combination of Lean and Six Sigma allows ISs drives a better business performance.

Lean Thinking has been introduced by James P. Womack firstly in the book "Lean Thinking: Banish Waste and Create Wealth in Your Corporation" [Womack and Jones 1996]. In this book, Womack and Jones address the revolution in manufacture has made by Toyota Production System (TPS). They suggest that a lean way of thinking allows companies to specify value, organize value creating activities in the best sequence, and improve the efficiency and performance of these activities. This book follows Womack's previous highly successful book "The Machine That Change the World" [Womack et al., 1990]. Both books address the TPS leads a Lean System and define, the five principles of Lean Thinking: specify Value, identify the Value Stream, make the Value-creating steps Flow, Pull, and pursue Perfection.

With the development of IT, Information Systems (ISs) such as ERP, Customer Relation Management (CRM), Product Data Management (PDM) and Inventory Management System (IMS), etc. can play a vital role in enabling an enterprise to achieve Lean production. [Powell et al., 2013] suggests that Enterprise Resource Planning (ERP) and Lean implementation process can be combined to develop a single "best-practice" process. Improving information management involves the integration of the IS infrastructure. The infrastructure of an organization generally consists in a large number of ISs componests that support various individuals, groups, departments and processes across the entire organization, and even multiple sites. This complicated system of inter-related elements ultimately needs to function as a whole in order to best support the organization.

Despite some similarities, Lean ISs have particular characters compared with Lean production processes. To identify ISs' wastes involves considering

the differences between ISs and production process. The following table shows the principles of achieving Lean in ISs and Production processes:

Table 2 *Comparison of Lean ISs and Lean Production systems*

	Lean ISs	**Lean Production**
Value	Defined by internal and external information consumers (business requirements)	Defined by ultimate customer
Value stream	A set of activities to produce the information for business needs	A set of activities to produce a product to customer
Flow	Making activities along value stream smooth and information synchronization. Avoiding conflicts and data inconsistencies	The progressive achievement of activities along the value stream to avoid stoppages, scrap or backflows during the production
Pull	A non-invasive system to avoid redundant information, business requirements trigger the activities during information value chain	A system to avoid unnecessary inventory, nothing is produced by the upstream supplier until the downstream customer signals a need.
Pursue perfection	No end to the process of reduce waste, and continuously increase the value of information stream	No end to the process of reducing wastes

The position of Lean development is that identification of and concentration on value, reduction of waste, and continuously improvement of processes, enables "business viability in a globally competitive and informed environment" [Millard 2001].

Moreover, Lean is achieved through a set of mutually reinforcing practices, including just-in-time (JIT), total quality management (TQM), total productive maintenance (TPM), continuous improvement, design for manufacturing and assembly (DFMA), supplier management, and effective human resource management. Additionally, Lean Thinking focuses on analyzing and optimizing value streams, the core philosophy of Lean Thinking can be summarized as Maximize Value, Minimize Waste. Use the least amount of effort, energy, equipment, time, facility space, materials, and capital while giving customers exactly what they want. Lean approach has been widely used beyond manufacture field, now non-manufacturing sectors are jumping onto the Lean bandwagon. Such as [Cusumano and Nobeoka 1998] in this book "Thinking Beyond Lean" look at applying lean principles to product development processes; [Myerson 2012] describes a Lean implementation methodology with

critical success factors in supply chain and logistics management, in this book, the author presents the Lean implement model "SCOR: Plan, Source, Make, Deliver and Return" which is a way to identify lean opportunities in the supply chain and applyLean thinking in construction domain by taking into account the particular problems in construction domain then proposing a dynamic model of performance improvement process.

Waste identification is the root for achieving lean, as well as the lean information is the foundation for lean business. In general, the notion of waste within the context of information management can be considered to include the additional actions and any inactivity that arise as a consequence of not providing the information consumer immediate access to an adequate amount of appropriate, accurate and up-to-date information. This concept is again analogous to the principles of lean thinking in a manufacturing context. It therefore follows that the philosophy of lean within the context of information management, is to identify and enable focused improvements on the various aspects of information management in order to eliminate waste and improve the flow of value. Compare and contrasted with the seven traditional wastes (viz. Overproduction, Waiting, Transport, Extra processing, Inventory, Motion and Defects) [Womack and Jones 1996] associated with manufacturing systems, regarding ISs have their particular characters, ISs' waste and challenges of building a lean IT and business collaboration from some case study [Scherrer-Rathje and Boyle, 2009]can be concluded as shown Table 3:

Table 3 *Challenges and Motivations of Achieving Lean ISs*

	Challenges of achieving Lean ISs	**Motivations of our research**
1	Failure demand, failure communication between IT and Business [Hicks 2007]	It requires a communication and collaboration processes to: 1) formalize business requirement into precise technical understandable demands; 2) Transform technical results to business understandable results.
2	Flawed flow includes the unnecessary or inappropriate activities [Hicks 2007]	A IS governance approach is necessary to implement Lean thinking to ISs,
3	Lack of a precise understanding for information quality [Millard 2001] [Bauch 2004]	It requires defining quality of ISs precisely and understanding the business needs clearly
4	Lack of a visible management commitment [Crandall and Coffey, 2005]	It requires formulating the agreements include precise and clear responsibilities, obligations and

		penalties, etc.
5	Implementing Lean thinking in enterprise, it requires to develop formal mechanisms to encourage and enable autonomy [Schonberger, 2005]	Lean thinking culture needs to be integrated in enterprise management. A formal automatic mechanism should be implemented to manage and govern the quality of business outcomes, performance of information and business value chain, and other critical factors.
6	Communicate lean wins from the outset [Alukal, 2003] and continual evaluation during the lean effort is critical [Liker 2004] [Crute et al., 2003].	It requires building long-term goals for sustainability of lean ISs to fit dynamic business requirements.

The success of Lean Thinking is based upon the reduction of wastes. The first important step for lean improvements is thus the identification of wastes, and then their removal, as well as a continuous performance improvement strategy is a critical success factor for application of Lean thinking. Six Sigma as a successful performance improvement approach has been widely used into various business fields, and it involves the rigorous pursuit of learning, problem-solving, process improvement, and ultimately, better business performance. Therefore, combine Six Sigma quality with Lean efficacy is not just a business improvement methodology that aims to maximize shareholder value by improving quality, speed, customer satisfaction and costs, but it is also an improvement strategy for IT and other aspect of enterprises. Lean Six Sigma refers to a more intelligent management of an organization, which first takes into account customers' requirements and their satisfaction by using data and facts for elaborating medium and long term strategies [Pamfilie 2012].

2.1.3 Six Sigma Approach for Information System

The Six Sigma methodology was created by Motorola in the mid-1980s. Six Sigma has been defined by practitioners and academic articles in a variety of ways. Key elements of the Six Sigma approach include a clear focus on the customers' needs, the use of performance metrics, a focus on improving business processes often through the reduction of inherent variation in the processes, clearly defined process improvement specialist roles, the use of data-driven and highly structured problem solving methodologies, and ultimately the generation of tangible business results [Schroeder et al., 2008]. [Kwak and Anbari 2006] identified two sources for Six Sigma processes:

1) Statistical field: the origin of Six Sigma comes from statistics and statisticians [Hahn et al., 1999]. According to this vision the six sigma method is a very rigorous quality control concept where many organizations still performs at three sigma level which was applied only to manufacturing processes, and it stipulated that a "capable" process was one that had a process standard deviation of no more than one-sixth of the total allowable spread.

2) Business field: Six Sigma is defined as a business strategy used to improve business profitability, operations' effectiveness and efficiency, to strengthen customers' satisfaction. As such Six Sigma requires the process standard deviation be no more than one-twelfth of the total allowable spread.

A traditional Six Sigma system is based on DMAIC (Define-Measure-Analyze-Improve-Control) methodology. DMAIC is a closed-loop process that eliminates unproductive steps, often focuses on new measurements, and applies technology for continuous improvement. With the development improvement of emerging technologies and business models, a new improvement methodology DMADV (Define-Measure-Analyze-Design-Verify) is generally used [Easton 2012]. The assumption of DMADV is the outcome of the entire process will be improved by reducing the variation of multiple elements. DMADV is a structured problem-solving framework derived from DMAIC, but intended to better align with design and development activities [Pyzdek and Keller, 2009].

Applying Six Sigma approach to IT and business collaborative environment, may increase the overall performance of the enterprise will be improved. Improving all of a department's individual processes could actually have a detrimental effect on the entire enterprise's ability to satisfy the customers' needs and provide product and services at the right time at the lowest cost.

Using the DMADV methodology to implement Six Sigma improvement approach in IT includes five steps:

- Define: Design the goals of information flow that are consistent with business goals and organization strategy. We assume that the "end user" of information flows are the Business Process. Therefore, the output of information flows should always satisfy the business processes' needs.

- Measure: Identify the Critical Success Factors which can indicate the performance of information flow and outputs, as well as identify information flow process' capabilities and risks. Then based on these identifications the quality of information flow can be measured precisely.

- Analyze: Analyze all improvement designs to select the best information system improvement design.

- Design: Optimize the design by simulating business situations to specify information system improvement design.

- Verify: Verify information system improvement design into business process.

Various literature reviews and discussions with six sigma leaders in organizations that adopted the six sigma method, such as [Swink and Jacobs 2012] that addresses Six Sigma adoption on operating performance impacts and contextual drivers of success. [Lee and Choi, 2006] makes a survey study about Six Sigma management activities and their influence on corporate competitiveness. [Johnson and Swisher, 2003] provided some implementation tips on Six Sigma application for R&D field, to implement a successful Six Sigma for IS. Based on these discussions we identify some key elements and issues we should pay attention on:

- Sustainable and visible management: It should fit organizational commitment: Six Sigma is not only a technique but also a philosophy business strategy that improve quality of organizations. Applying of Six Sigma to the IS field, involves that the strategy of IS should always fit the organizational business strategy, that IS management should always participate organizational management. Therefore, designing IS Six Sigma processes should support the business objectives.

- IT and infrastructure selection, management and control skills: from business perspective, implementing a successful Six Sigma requires continuing education and training of managers and participants. Six Sigma for IS not only requires continuing training IT specialists but also a continuous updating of IT and IS infrastructures. For example, transferring IS infrastructure to Cloud environment to take advantage of cloud computing promised low cost, more convenient. However, if we cannot use the emerging business paradigms and new technologies appropriately, business and IS cannot get benefit from them. To this end, we should design IS monitoring strategy to grantee IS performance and make sure the new technologies and paradigms are beneficial assets for organization.

- Set clear expectations, audit performance and report: To set clear objectives and expectations, customized performance metrics and coping mechanism for unexpected results, we should monitor ISs' state based on these expectations and metrics, any unexpected state should be detected and ISs should be able to response to the detected unexpected state according to designed coping mechanism.

Six Sigma approach for information system aims at improving the quality of information flow and at making information flow be able to produce satisfied output for business processes' needs. Clear customer-oriented metrics, appropriate measurements, match organizational objectives and performance monitoring processes are critical success factors to implement Six Sigma in IS. The integration of Lean and Six Sigma means to produce information flow to fit

business processes with maximum value and minimum cost, allowing business processes to match enterprise-level's agility strategy.

2.1.4 Conclusion

Enterprise modeling methods are designed to integrate different points of view to support industrial organization and decision making. However, the different modeling methodologies are static and lack of taking into account quality assurance. To deal with the collaboration of internal and external enterprises and to meet the continuous improvement requirements require to define a new modeling approach which can allow to specify and monitor the industrial system and its associated decision system. Furthermore, the increasing importance of information systems involve to take into account the information value-chain performance to enhance organizational profitability.

Applying Lean Six Sigma to information system environment is a way to improve the performance of information value chain and optimize the organization of ISs, so that quality of ISs drives business agility.

Moreover, applying Lean Thinking to ISs involves following the five principles:

1. Identify ISs' value, Due to IS focuses on supporting business processes, regarding information, the value of information is if it increases the customer's value and if it is used to assure quality, speeds up the process, or reduces cost;

2.Create value stream, specify value-added activities and spot the activities cannot add value to business processes, any activity in information value chain that leads to unused information for business process is waste;

3. Progress value-creating steps flow, smooth the information flow;

4. Comply with cascading production and delivery system, only produce when business processes signal a need;

5. Apply Six Sigma approach to continually pursue perfection.

The main limit of applying Lean to ISs is that it lacks of integration of Lean business view of information technical value flow. Adding information technical value to business value flow according to business objectives is the main challenge for applying Lean to ISs.

In a summary, the first important step for lean improvements is thus the identification of waste, and the second its removal. In our IT collaborative business environment, ultimate essential concern is the quality of information has to fit the business requirements and add value to business process without any non-value activity and extra cost during the information flow. Ideal IS provides high quality and precise information resource for business processes, however, to deliver tangible business value, business processes need to swiftly

adapt its strategies to reflect IT changes. Business processes and IS should be aligned to achieve enterprise-level business agility.

The following of State of the Art is organized in different sections. In section 2.2 we introduce the combination of Business Process Management (BPM) and Enterprise Architecture (EA) as an efficient way to achieve business agility. After that, in section 2.3 we introduce SOA as an architectural style well suited for modern EA. Its loosely coupled characteristics allow enterprises to achieve business and IT alignment. It brings new solutions for improving agility of ISs and business processes, thereby enabling delivering better business value. After talking about the benefits from combination of BPM, EA and SOA, to guarantee those promised benefits, we are still facing some challenges, such as guarantee Quality of Service, elicitation and refinement of Non Functional Requirements, Management of Service Level Agreement (SLA), security, minimize costs, maximize utilization, correct mistakes, etc. To overcome these challenges, in section 2.4 we introduce the performance measurement with autonomic management strategy. It aims at monitoring business situation without interrupting the ongoing business processes. However, existing solutions are not dynamic. They cannot fit the emerging new business paradigms nor fit implementation elasticity and scalability involved by large scale collaborative business processes. In section 2.5, we discuss Cloud Computing's characteristics and cloud monitoring challenges. Cloud Computing dramatically changes the way business is run. Lowering operation cost, highly scalable, easy access, reducing business risk and maintenance expenses, make cloud computing attractive to business participants. However, gaining these promised benefits from cloud requires new performance measurement and monitoring solution for organizations. There are still some open issues that need to be solved for building cloud-ready monitoring and governance. Finally, State of The Art will end up with a chapter conclusion (section 2.6) presenting existing limits.

2.2 Combining Business Process Management and Enterprise Architecture

2.2.1 Introduction

Enteprise modeling mothodolgies provide ways to support integration and decision making. However, they lack of flexibility and do not take into account any governance / quality monitoring steps. Moreover, in order to add more value picked from information systems to business process value chain requires to align information systems with business and to improve business process quality.

We define a business process as a complete, dynamically coordinated set of activities or logically related tasks including required implementation elements that must be performed to deliver value to customers or to fulfill other organizational strategic goals. In our research work, all business process implementation elements including Business Tasks, Services, Infrastructure elements (such as hardware, equipement, database, software, etc.) are defined as Business Resources.

From a business perspective, enterprise needs to design an agile Business Process Management (BPM) to manage all resources and activities during all the business processes to maximize the benefit of agile ISs. From a technological perspective, the enterprise needs to establish a platform that enables the appropriate collaboration by creating visibility, traceability, and integrity between targets and solutions throughout all roles and tools [Jensen et al., 2011]. Enterprise Architecture (EA) field initially began to address two problems [Sessions 2007]:

- System complexity: realizing the importance of IT/IS, enterprises were invest more and more money building ISs;

- Gap between IT and business: enterprises were finding it more and more difficult to align those increasingly investment of IT aligned with business need.

Enterprise Architecture (EA) gives a platform for BPM to implement enterprise's business strategy. BPM provides the business context, understanding and metrics. EA provides the discipline for translating business vision and strategy into architectural change. In this section we analyze and compare four top EAs and summarize existing challenges for a success EA, then talk about BPM, and finally introduce the convergence of BPM and EA is an efficient way to achieve business agility.

2.2.2 Enterprise Architecture Framework

In order to manage the increasing complexity of information technology systems and to deliver maximum real business value, Enterprise Architectures (EAs) have been developed for decades. EA provides a clear vision of the organization's current IT assets and business processes. EA frameworks usually provide a context in which all stakeholders in an organization can communicate and collaborate to set their enterprise architecture. EA as a discipline provides the foundation for an organization to align strategic objectives with opportunities for change. This is achieved through portfolio gap analysis, transition planning, and architectural governance. The resulting enterprise architecture is used to identify impacts of changes on the enterprise and to drive the gap analysis between current and future states.

Definition for EA is presented by [Lankhorst M. et al., 2005] as "enterprise architecture is a coherent whole of principles, methods and models that are used in the design and realization of an enterprise's organizational structure, business processes, information systems, and infrastructure." Enterprise architecture explains how all the information technology elements in an organization – systems, processes, organizations', and people – work together as a whole [Morganwalp and Sage, 2004]. [Hämäläinen and Liimatainen, 2008] presents that EA commonly has four viewpoints: business architecture, information architecture, application architecture and technology architecture. These viewpoints are promoted in many widely used frameworks. Enterprise Architecture Framework (EAF) defines how to organize the structure and views associated with an EA. There are three components of EAF: Views, Methods, Training/Experience.

In 1987, [Zachman, 1987] introduces his Framework for Information Systems Architecture which is commonly accepted as the first approach towards the discipline of EA. EA is characterized by the use of frame-works that support the analysis of the enterprise from the business-level down to the IT-level. Five years later, in 1992, [Zachman and Sowa, 1992] enhances Zachman framework. This framework introduces the basics of EA. In 1996, the Clinger-Cohen Act [EAO 2004] (formerly known as the Information Technology Management Reform Act) of the U.S. government directed federal agencies to implement a holistic approach to align information technology to their business goals. According to [Malveau, 2004], this led to the creation of the term enterprise architecture. Since then, the amount of interest devoted to EA has increased. Today EA is well-known as a hierarchical approach to align Business and IT. One of the most popular frameworks, inspired by the Zachman framework, is The Open Group Architecture Framework (TOGAF) defined by the Open Group [TOGAF, 2003].

According to [Leganza, 2004], EA has developed two major approaches: a top-down approach that assumes comprehensive scope and strictly follows a formal process, and a bottom-up approach that starts with infrastructure technology standardization and then moves up the target high-priority problem areas and eventually influence business architecture. Most of the EA methodologies consider four organizational levels (see Figure 3):

- Business layer: The business architecture represents the fundamental organization of the corporate (or government agency) from a business strategy view point. Design and evolution principles for business architecture can be derived e.g. according to the market based approach or the resource based approach to strategic management.

- Data layer presents the collection, organization, and distribution of data. The fundamental organization of information system is analyzing data coupling.

- Application layer: The software architecture represents the fundamental organization of software artifacts, e.g. software services and data structures. A wide range of design and evolution principles from computer science is available for this layer.

- Technology (or infrastructure) layer: The technology architecture represents the fundamental organization of computing/ telecommunications hardware and networks. A wide range of design and evolution principles from computer science is available for this layer too.

Each level describes either what currently exists (as-is) or what should exist (to-be). The study of Braun and Winter [Braun and Winter, 2007] mentiones that EA as a hierarchical approach usually applies the 'IT follows business' principle, starting with strategic positioning from the business management point of view, then deriving appropriate organizational processes and structures on this basis, and finally specifying the information system, i.e. the interaction between human and technical information system components that appropriately support business requirements.

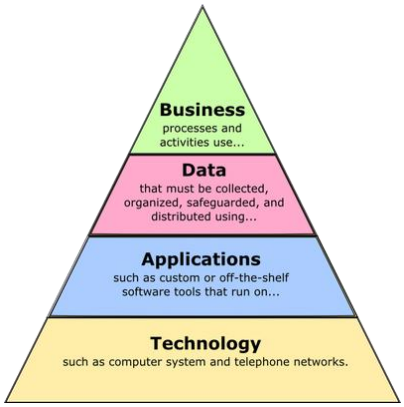


Figure 3 *Multi-Layer of EA (http://www.records.nsw.gov.au/recordkeeping/government-recordkeeping-manual/guidance/recordkeeping-in-brief/images-1/layers.png/view)*

2.2.3.2 Zachman Framework

Zachman pointed out the challenge to manage the complexity of increasing distributed systems. [Zachman 1987] sets that business value and agility could best be realized by a holistic approach of systems architecture that explicitly looked at every important issue from every important perspectives. Zachman originally described its work as an information system architectural

framework. It was soon to be renamed as an enterprise architecture framework. Zachman's multi-perspective approach has profound impacts on subsequent researches.

The Zachman's Framework relies on the theory that there exists a set representation (models) for describing, designing and building complex objects. The Zachman "Framework" is a taxonomy for organizing architectural artifacts. It takes into account both the artifact targets and the particular issue which is addressed. This framework draws two distinct classification systems to define the set of representations that are needed to manage the complexity and change of these objects. This set of representations forms the cells of the framework.

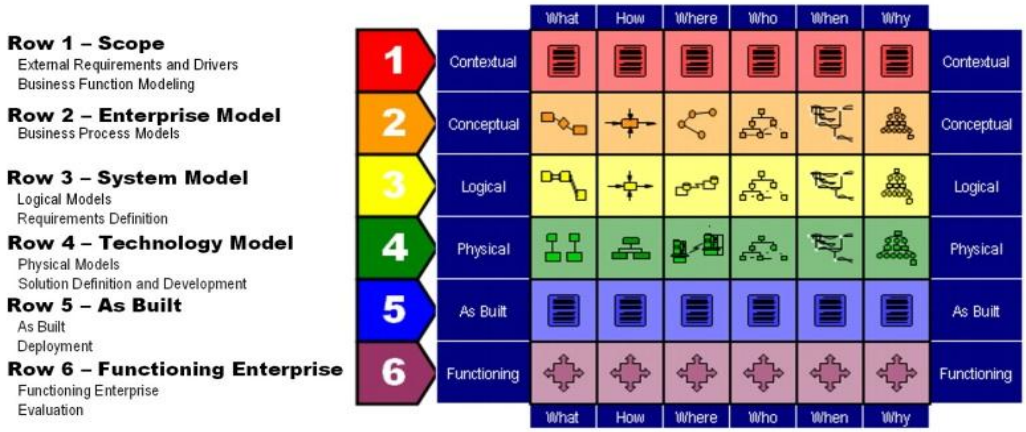


Figure 4 *Zachman Framework (from zachmaninternational.com/s/Zachman_Framework.asp)*

The first classification consists in the six fundamental questions (commonly used in journalism) associated to columns of the framework: who, what, when, where, why, and how. This set of questions has been used for millennia to describe situations or objects and forms the basis of human communication.

The second classification is based on the stakeholders from traditional architecture: owner, designer, builder. It has been extended to include: planner and subcontractor. This set of perspectives has been used for centuries in the architectural field to engineer in a way of independent of geography, culture, language, politics or technology. These perspectives form the rows of the framework [Zachman, 2003].

Since the 1990s, the Zachman Framework has been widely used as a mean of providing structure for Information Engineering-style enterprise modeling. The Zachman Framework can be applied both in commercial companies and in government agencies. It can be applied to an entire governance agency or corporate at an abstract level, or it can be applied to various departments, offices, programs, subunits and even to basic operational

entities. Enterprises should pay attention on different points when they are using the Zachman framework:

- Each artifact should be in one and only one cell;
- Only when every cell in that architecture is complete, the Zachman grid can be considered a complete architecture;
- Cells in columns should collaborate with each other related to each other.

However, in many cases, Zachman framework by itself is not a complete solution. In complex situation, it does not give a step-by-step process for creating a new architecture. It does not even give an approach to show a need for a future architecture. To this end, we need to look for other methodology frameworks.

2.2.3.3 The Open Group Architecture Framework

TOGAF has been defined and is owned by The Open Group. TOGAF divides an EA into four categories [TOGAF 2003]:

- Business architecture: describes the processes the business uses to meet its goals;
- Application architecture: describes how specific applications are designed and how they interact with each other;
- Data architecture: describes how the enterprise data storage is organized and how these data can be access and organized;
- Technical architecture: describes the hardware and software infrastructures that support and interact with applications.

TOGAF complements the work previously introduced by Zachman. Whereas Zachman Grid describes how to categorize EA artifacts, TOGAF describes the process to categorize EA artifacts. The most important part of TOGAF is the Architecture Development Method (ADM) (see Figure 5). ADM describes an architecture process to categorize EA artifacts. It is applied to develop an enterprise architecture which will meet the business and in-formation technology needs of an organization. It may be tailored to the organization's needs and is then used to manage the execution of architecture planning activities.

[Perks and Beveridge, 2003] suggests that TOGAF provides only prescriptive document templates – merely for inputs and outputs, and does not specify exactly the different documents.

TOGAF is more flexible than Zachman's framework because it allows phases to be done incompletely, skipped, combined, reordered, or reshaped to fit the needs of the situation. However, TOGAF does not guarantee that the generated EA is convenient as results are dependent on the experience of TOGAF execution staff: it is a big risk for enterprises.

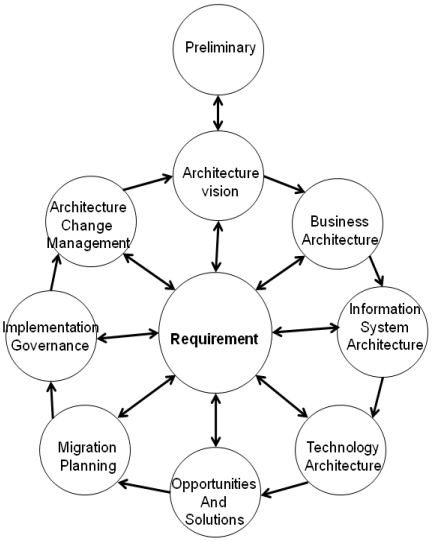


Figure 5 *ADM of TOGAF (from The Open Group: http://pubs.opengroup.org/architecture/togaf8-doc/arch/)*

2.2.3.4 Federal Enterprise Architecture

The Federal Enterprise Architecture (FEA) is a more complete methodology compared to the previous two. FEA has both a comprehensive taxonomy, like Zachman, and an architecture process, like TOGAF. FEA is latest attempt by the US federal government to unify its myriad of agencies and functions under a single common and ubiquitous enterprise architecture.

A full treatment of FEA needs to include all of the following:

i. A perspective on how enterprise architectures should be viewed;

ii. A set of reference models for describing different perspectives of the enterprise architecture;

iii. A process for creating an enterprise architecture;

iv. A transitional process for migrating from a pre-EA to a post- EA paradigm;

v. A taxonomy for cataloging assets that falls in the purview of the enterprise architecture;

vi. An approach to measure the success of using the enterprise architecture to drive business value.

In order to give standard terms and definitions for the domains of EA and thereby facilitate collaboration and sharing across the federal government, FEA proposes five reference models [FEA 2005]:

i. The Business Reference Model (BRM) is a standardized framework to measure the performance of major IT investments and their contribution to the program performance.

ii. The Components Reference Model (CRM) is a business reference model. This function-driven framework describes the business operations of the Federal Government in an independent of the agencies that perform them. The BRM is the first layer of the Federal Enterprise Architecture and it is the main viewpoint for the analysis of data, service components and technology.

iii. The Technical Reference Model (TRM) is a business and performance-driven, functional framework that classifies Service Components with respect to how they support business and/or performance objectives.

iv. The Data Reference Model (DRM) describes the data and information that support government program and business line operations. This model enables agencies to describe the interaction and exchanges between components.

v. The Performance Reference Model (PRM) is a component-driven, technical framework categorizing the standards and technologies to support and enable the delivery of Service Components and capabilities. It also unifies existing agency TRMs and E-Government guidance by providing a foundation to advance the reuse and standardization of technology and Service Components from a government-wide perspective.

The framework uses assessment criteria to evaluate the performance and effectiveness of agency enterprise architecture programs. Each criterion consists in five performance levels, scored from 1-5. Related assessment criteria are grouped into three capability areas [FEA 2009].

- Completion addresses the following Key Performance Indicators (KPIs): Target Enterprise Architecture and Enterprise Transition Plan; Architectural Prioritization; Scope of Completion; and Internet Protocol Version 6 adaption.

- Use addresses the following Key Performance Indicators:

1) Performance Improvement Integration: Measures how effectively the agency has aligned its performance improvement plans with its enterprise transition plan.

2) Capital Planning and Investment Control integration: Measures the alignment between the enterprise transition plan and the agency;

3) FEA Reference Model and Exhibit 53 Part Mapping: Measures the completeness and accuracy of the primary FEA reference model mapping and specification of the IT investments in the agency IT portfolio.

4) Collaboration and Reuse: Measures agency progress in migrating their target applications and shared services portfolio, and creating a services environment within the agency.

5) EA Governance Program Management, Change Management and Deployment: Measures the degree to which the agency governs and manages the implementation and use of EA policies and processes.

- Results address the following Key Performance Indicators: Mission Performance, Cost Savings and Cost Avoidance, Measuring EA Program Value.

As the following figure shows (see Figure 6) the Federal Enterprise Architecture Practice Guidance has defined three types of architecture [FEA 2007]:

1) Enterprise architecture,
2) Segment architecture,
3) Solution architecture.

These different architectures address different concerns from different business perspectives. The process of FEA is composed of different steps: Analyze architecture, Define architecture, Design investment and funding strategy, manage and execute projects.

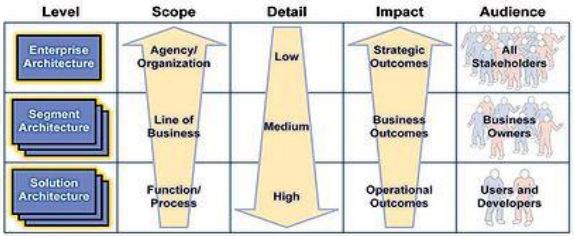


Figure 6 *FEA: Architecture Level Attributes (from en.wikipedia.org/wiki/Federal_Enterprise_Architecture)*

2.2.3.5 Gartner Framework

Gartner methodology is different from the previously presented EA methodologies. Gartner is one of the best known IT research and consulting organizations in the world and its Enterprise architecture is focused on strategy, and not about engineering. The two things that are most important to Gartner are where an organization is going and how it will get there [Sessions 2007].

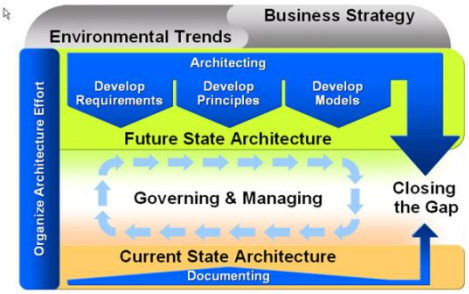


Figure 7 *Structure of Gartner EAF (from iea.wikidot.com/gartner-ea-framework)*

Gartner believes that EA is about bringing together three constituents: the business owners, the information specialists and the technology implementers. EA must start by identifying where an organization is going and not with where it is.

The complexity and diversity of business processes are big challenges for most organizations. Gartner recommends that an organization should have clear and simple ideas about where its strategic direction is heading. This aims at making sure that everybody understand and share a single vision. When an organization has this single shared vision of the future, the organization can efficiently orchestrate business resources and complement changes to keep investments be beneficial for business value.

2.2.3 Challenges for a Successful EA

	Rating			
Criteria	Zachman	TOGAF	FEA	Gartner
Taxonomy Completeness	******	**	**	*
Process Completeness	*	******	**	***
Reference Model Guidance	*	***	******	*
Practice Guidance	*	**	**	******
Business Focus	*	**	**	******
Success Measurement	*	*	***	***
Governance Guidance	*	**	***	**

Note: * Means it does a very poor job in this field;
****** Means It does a very good job in this field;
The more * means a better job in the field

Figure 8 *Comparison of Top Four EAFs*

These EA frameworks are very different in their approaches. They focus on different concerns and each of them has its own strengths and weaknesses. We compare and score these four EAFs according to 7 criteria (see Figure 8):

These criteria cover the design, implementation and evaluation of building EA for organizations.

1) Taxonomy Completeness refers to how well we can use the methodology to classify the various architectural artifacts. This is the entire focus of Zachman. None of the other EAs focus as much on this aspect.
2) Process Completeness refers to how fully the methodology provides a step-by-step process for creating an EA. This is almost the entire focus of TOGAF. None of others focus as much on this aspect.

3) Reference Model Guidance refers to how useful the methodology is in building a relevant set of reference models. This is almost the entire focus of FEA. Others pay less attention on this aspect.
4) Practice Guidance refers to how much the methodology helps assimilate the mindset of EA into an organization and develop a culture of agility. This is a primary focus of Gartner's architecture practice. Other EAs do not pay much attention on this aspect.
5) Business Focus refers to whether the methodology will focus on using technology to reduce expenses and to increase business income. This is another focus of Gartner, while others do not focus on this aspect.
6) Success Measurement refers to how well the methodology will ensure organizations build a successful EA. None of these EAs provide a satisfied solution.
7) Governance Guidance refers to how well the methodology will help organizations build an effective governance model for maintaining the successful of EA. None of these methodologies focuses on this aspect.

According to the comparison of these leading EA frameworks, we can see that each of them focus on individual aspect, none of them covers all the criteria. However, we lack of a good choice for Success Measurement and Governance Guidance, as none of these methodologies provides a measurement for organizations to check whether they build a successful EA or not. Furthermore, none of them provides an effective governance guidance for organizations to govern their EAs' status. Consequently organizations may lack of understanding their EAs' development status and their organizations' running status. Therefore, this blindness might bring unexpected results to both technology and business sides. This comparison motivates our research to propose a new methodology to cover the existing limits and challenges and pay attention on agility of IT and business.

For many organizations, the best choice is to understand the value of each methodology and pick the most required strengths from these methodologies and blended together to be beneficial for both business side and technology sides. [Sessions 2007] suggests that building a successfully enterprise architectural should include following objectives:

- Improve using IT to drive business adaptability;
- Support a closer partnership between business and IT groups;
- Improve focusing on organizational goals;
- Improve understanding a direct correlation between individuals work and the organization's success;

- Reduce numbers of failed IT systems, reduce complexity of existing IT systems and improved agility of new IT systems;

- Provide a closer alignment between IT deliverables and business requirements.

Nevertheless, an EA methodology has no value unless it delivers real business value as quickly as possible and federates both business side and IT side to make them work effectively toward the same goals. Even EAs has been evaluated so long, it also has to face two new major problems.

- Managing the increasing complexity of IT systems.

- Taking into account increasing difficulty in delivering real business value with those IT systems.

An excellent EA governance framework is a critical tool to ensure that EA matures in a competency enhancing fashion that enables both the business and IT strategies. To this end, a well-functioning EA Governance is necessary to support EA and to achieve a successful IT organization. Appropriate governance methods enable IT to become a key differentiator in creating an agile, adoptable enterprise, and to enable business process to be adapted continuously. These requirements are necessary to support enterprises to get the high flexibility to meet the new market conditions. The development of a proactive governance of each architecture discipline is critical to impact the enterprise architecture strategy, because only high flexible companies could survive in the long term.

[Weill and Ross 2004] have presented that even some organizations have noticed the importance of EA Governance, most of the EA Governance methods separate the IT governance from business-performance metrics. There is still a big gap between business requirements and IT technology capacity. It is difficult to make them understand each other [Becker, et al. 2004]. The challenges for EA governance are:

- How are the business principles translated to IT principles?

- What are the desirable IT behaviors?

- How Technology choices will guide the enterprise's approach to IT initiatives?

- What is the plan for keeping underlying technologies up-to-date?

- Which infrastructure services are critical to achieving the enterprise's strategic objectives?

- What are the market and business process opportunities for new business applications?

Enterprise Architectures (EA) have some similarities with Enterprise Modelling (EM) methodologies. The main purpose of EA and EM is to compose different components of enterprise and to support collaboration of different enterprise aspects to achieve common organizational goals. EA and EM are also sharing some limits such as lack of dynamicity and quality assurance. To

overcome existing limits and challenges our research aims at proposing a new methodology which will not only guide an organization build customized EA but also provide governance model to evaluate the performance of EA and quality of business outcomes. This methodology should pay attention on the critical success factors for EA and to ensure IT investment benefit business goals. With the three primary goals (portability, interoperability and reusability) Model Driven Architecture (MDA) is a positive way to organize and manage enterprise architectures, which supported by automated tools and services for both defining the models and facilitating transformations between different model types [OMG 2003].

Improving business performance, we should fully understand the challenges of Business Process Management (BPM). In the following section we will introduce the challenges of BPM, and the combination of BPM and EA.

2.2.4 Business Process Management

Business Process (BP) has been defined in different ways according to various researches. [Hammer and Champy 1993] defines Business Process as "a collection of activities that takes one or more kinds of input and creates an output that is of value to the customer. A business process has a goal and is affected by events occurring in the external world or in other processes".

In [Davenport 1993] book, a business process is defined as "a structured, measured set of activities designed to produce a specified output for a particular customer or market. It implies a strong emphasis on how work is done within an organization, in contrast to a product focus's emphasis on what. A process is thus a specific ordering of work activities across time and place, with a beginning, an end, and clearly identified inputs and outputs: a structure for action."

[Ko, 2009] concludes previous works by providing two main dimensions of business processes: 1) The Level dimension includes three levels of management activities (viz. operational control, management control and strategic planning). 2) The Core competency dimension includes three groups (viz. core business processes, management business processes and support business processes). Then it defines a business process as "a series or network of value-added activities, performed by their relevant roles or collaborators, to purposefully achieve the common business goal."

[Škrinjar and Trkman 2013] defines a business process as a complete, dynamically coordinated set of activities or logically related tasks that must be performed to deliver value to customers or to fulfill other strategic goals.

According to the BPM survey of [van der Aalst, et al. 2003], BPM is defined as "Supporting business processes using methods, techniques, and

software to design, enact, control, and analyze operational processes involving humans, organizations, applications, documents and other sources of information." This survey suggests bringing together (computer) scientists and practitioners in order to work on advanced BPM methods, techniques, and tools.

[Papazoglou and Heuvel, 2007] summaries that the term management of business processes includes process analysis, process definition and redefinition, resource allocation, scheduling, measurement of process quality and efficiency, and process optimization. Process optimization includes collection and analysis of both real-time measures (monitoring) and strategic measures (performance management), and their correlation as the basis for process improvement and innovation. BPM promises the ability to monitor both the state of any single process instance and all instances in the aggregate, using present real-time metrics that translate actual process activity into key performance indicators.

[J2CA 2003] suggests that BPM is a commitment to express, understand, represent and manage a business in terms of a collection of business processes that are responsive to a business environment of internal or external events. [Bajwa, 2011] defines BPM as: A strategy for managing and improving the performance of a business through continuous optimization of business processes in a closed-loop cycle of modeling, execution, and measurement

Based on these definitions, in our research we adapt a definition from [Trkman 2010]. A business process is a complete, dynamically coordinated set of activities or logically related tasks including required implementation elements that must be performed to deliver value to customers or to fulfill other organizational strategic goals. All business process implementation elements including Business Tasks, Services, Infrastructure elements (such as hardware, equipement, database, software, etc.) are defined as Business Resources in our research. We take BPM as a solution: 1) to synchronize the communication among all activities and resources along business value chain and 2) to manage these activities and resource to achieve business goals, 3) to continuously improve business performance by applying appropriate tools. BPM should translate a firm's strategy into specific needs and enable the execution of the strategy. BPM refers to aligning processes with the organization's strategic goals. BPM follows initiatives established throughout the 1980s and 1990s such as Total Quality Management (TQM), Business Process Reengineering (BPR) and Enterprise Resource Planning (ERP). These methodologies all strove to improve the performance of businesses through measurement, restructuring, automation, and supporting systems. BPM tools must be suitable for business analysts and fit between the business environment and business processes, as well as continuous improvement efforts to assure sustained benefits from BPM is also a critical success factor for BPM.

As information systems (as support systems for business) are playing key role in business process. The beneficial of adopting ISs with BPM can be summarized as follows: increased visibility and knowledge of enterprise's activities, increased ability to identify bottlenecks, increased identification of potential optimization, reduced the time to market, easy definition of duties and roles properly internal and external of enterprise. BPM is an efficient tool to auditing as well as monitoring business performance and outcomes. The measurement and evaluation of the efficiency of business processes is a very important facet of Business Process Management Systems (BPMSs) as it provides real-time feedback on the status of processes and measures the time and cost of processes so that they can be optimized. Performance of BPM can impact a service industry firm's performance through increased revenue, cost reduction, cycle-time improvement, increased customer satisfaction and improvements in any other metric considered as important for creating value [Vukšić et al. 2013].

In the following we introduce how BPM and Enterprise Architecture (EA) can be combined to gain additional benefits from BPM:

- Consume architectural considerations into BPM solution delivery;
- Enable reuse and IT governance;
- Provide corporate approved templates and blueprints to govern and facilitate BPM business process design;
- Optimize and deploy process models for maximizing business outcomes;
- Publish updated process for corporate and IT governance.

2.2.5 Combing of BPM and EA

Achieving business-IT alignment without an EA (e.g. business architecture) approach is not easy [Harry, et al., 2011]. It also would be hard to achieve process and activity alignment in an operational approach without BPM. [Rosing et al. 2011] concludes that achieving success of BPM requires a business focused EA, as well as building successful and business focused ISs requires a clear and accurate understanding of business strategy. Therefore, the combination of BPM and EA is the key to link business strategies and information technologies.

Both of BPM and EA can gain additional benefits from each other. EA can provide corporate approved templates and blueprints to govern and facilitate BPM business process design. It can optimize and deploy process models for maximized business outcome, as well as publish updated process for corporate re-use and IT governance. BPM can gain additional benefits from EA. It can reflect business processes for enterprise architecture analysis and blueprint

design. It also can analyze business processes to verify optimal IT implementation (data, applications, processes, systems, and technology). It can examine the impact of using processes in a company or in an inter-companies collaboration context. Lastly it can validate business objectives against other corporate solution delivery approaches. All in all, from an EA perspective, establishment of the proper business context is a prerequisite for effective planning of architectural change. Additionally, BPM projects are governed and guided by architectural considerations and targets, which can be provided naturally by EA. From a BPM perspective, process change can lead to the need for IT architecture change, which can be driven naturally by EA. EA can reference business processes for architectural analysis and design business processes which are naturally provided by BPM [Rosing et al. 2011].

2.2.6 Conclusion

In this section we introduced Enterprise Architectures (EA), Business Process Management (BPM) and how combining business process management and enterprise architecture frameworks can help organizations implementing complex IT solutions and aligning the IT with the business model, business process, application, information, and infrastructure domains that are all part of the enterprise architecture.

From the comparison of top four EAs we can see that each of them has its own strengths and weaknesses. None of them covers all the requirements, especially they lack attention on the Success Measurement Guidance and Governance Guidance. However, these two factors are critical to ensure established EAs can support BPM. BPM can be constrained by EA, to achieve agility of ISs and business. This situation motivates our research to propose a new solution which can provide Measurement and Governance for organization to govern and improve the performance of their EA and BPM, and then help organizations to achieve ISs and business agility.

However, EAs do not provide a clear solution to govern the established enterprise architecture, i.e. identify whether it matches the global business goals or not. BPM lacks the architectural principles, policies, and standards that emerge and develop during the link to strategy and business model and then through the architecture lifecycle. The combination of BPM and EA still requires strengthening interoperation, combining elements, and seamlessly communicating multi-layers and opening information system.

The requirement of combing BPM with EA fits the ability of service oriented computing brings new solution to combine BPM and EA. Service Oriented solutions build an abstract layer as a middleware between business and ISs. This aims at narrowing the gap between business and information system,

reducing the cost and risk, then to ultimately improve business agility. Service Oriented method provides an architectural solution which is well suited for modern EA. Furthermore, the value proposition of service oriented solution is centered on agile and aligned business and IT design and delivery, thereby enabling business integrity and operational excellence [Rosing et al. 2011]. Service oriented solution does not increase the size of IT systems, but it does increase their interoperability. With service oriented solution, the IT systems perform services that defined and described in the context of the enterprise's business activities. The major benefit of service oriented solution is it improves business agility and interoperability not only within enterprises, but also between enterprises. Service Orientation is part of the EA mainstream.

To this end, in the next section we will introduce the Service Oriented Architecture before paying attention on the way it can be governed.

2.3 Service Oriented Architecture

2.3.1 Introduction

Service-Oriented Architecture (SOA) shares the similar objectives goals to EA in respects to different views to identify enterprise concerns and interests. SOA addresses the requirements of loosely coupled, standards-based, and protocol independent distributed computing. It allows enterprises to seize business opportunities effectively and quickly. SOA is a style and/or component of EA rather than an alternative or a competitor [Erl 2007]. It maps enterprise information system appropriately to the overall business process flow [Papazoglou 2007].

Over the past few years, BPM and SOA have been advocated as evolutionary initiatives that will enable organizations become more agile through better flexibility and better reusability that reduces costs and increases efficiency. Coupling BPM and SOA facilitates the next phase of business process evolution, from merely automating repeatable processes to flexible automation of dynamic processes [Jasmine 2005]. When combined together, BPM and SOA become synergetic and provide the most favored infrastructural approach to counter the challenges imposed by changing business environment. They will require enterprises to implement BPM processes as services and BPM tools as service-oriented composition applications [Colleen 2006]. Processes modeled by BPM tools can be implemented by SOA more efficiently [Hilty, 2009]. The use of BPM in concert with SOA is the fast path to ensuring true business agility. BPM and SOA provide a perfect combination for enterprise

computing. BPM provides the higher-level abstraction for defining businesses processes, as well as other important capabilities of monitoring and managing those processes [Imran 2008]. SOA provides the application platform to bridges to the business processes and the operational resources [Bruce 2006].

In 2006, Forrester Research Inc. wrote that BPM and SOA markets are becoming one and converging to the point that the "integration suite" market category is obsolete and is being replaced by the emerging "Integration-Centric Business Process Management Suite" (IC-BPMS) [Ken and Henry 2006]. The adoption of a proper governance model should take into account the fact that a BPM-SOA initiative has to endorse a new state of mind that brings business and IT closer together than any previous time before. A successful implementation requires a strong harmony between the Business-driven process design and IT-driven architecture and applications design. It should also clearly articulate the business entity who will own (and be accounted for) the BPM-SOA initiative [Kamoun, 2007].

2.3.2 Essence of SOA

One of SOA's greatest strengths is its enhanced flexibility due to the selection composition and orchestration of services. It enables to set agile business processes support, loosely coupled with a specific implementation technology. SOA provides several significant benefits for distributed enterprise systems, such as interoperability, efficiency, and standardization [González and Ruggia, 2010]. SOA is designed to eliminate the barriers of distributed enterprises so that application integrate and run seamlessly. It facilitates businesses' collaboration and change. It allows developers to overcome many distributed enterprise computing challenges including application integration, transaction management, security policies, while allowing multiple platforms and protocols as well as leveraging numerous access devices and legacy systems [Alonso, et al., 2004].

[OASIS RM 2006] presents SOA as a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. Visibility, interaction and effect are key concepts for describing the SOA paradigm. Visibility refers to the capacity of providing descriptions for such aspects as functions and technical requirements, related constraints and policies, and mechanisms for access or response. Interaction is the activity of using a capability. It is typically mediated by the exchange of messages, an interaction proceeds through exchanged information and invoked actions.

Service is the key concept of SOA. A service is a mechanism used to enable access to one or more capabilities. The access is provided thanks a

prescribed interface and is exercised consistently with constraints and policies as specified by the service description. A service is provided by a Service Provider and used by a Service Consumer. Service description makes available critical information that a consumer needs in order to decide whether or not to use a service. A service description should include service reach-ability, functionality, related policies, and service interface.

From a dynamic perspective, there are three fundamental concepts that are important in understanding what is involved in interacting with services: (1) the visibility between service providers, and consumers (2) the interaction between consumers and/or providers, and (3) the real world effect of interacting with a service.

Visibility is the relationship between service consumers and providers that is satisfied when they are able to interact with each other. Preconditions to visibility are awareness (service provider and consumer should know each other existing), willingness (service participants are willing to engage in service interaction) and reach ability (service participants are able to interact).

Interacting with a service involves performing actions with the service. It includes an information model defining the information that may be exchanged with the service and behavior model which is the knowledge of the actions on, responses to, and temporal dependencies between actions on the service.

Real world effect can be the response to a request for information or a change in the state of some defined entities shared by the service participants

2.3.3 Multi-layer Organization

SOA has attracted attention for its promise of new ways to cope with IT architecture challenges. The promised benefits are alignment of IT with the business and maximal reuse of IT assets. It helps assuring that investment in expensive IT will result in lasting value to the business. So, [Mos et al. 2008] suggests that SOA can take a part of a Business Process Management (BPM) platform. It can also be an enabler for business agility through better adaption of IT resource to business needs. The authors defined 3 layers:

- Infrastructure layer which is related to the SOA infrastructure architecture definition;
- IT SOA layer which contains technical connotation elements;
- Business layer which corresponds to the high-level business oriented definition of the business process in design conceptual and runtime conceptual.

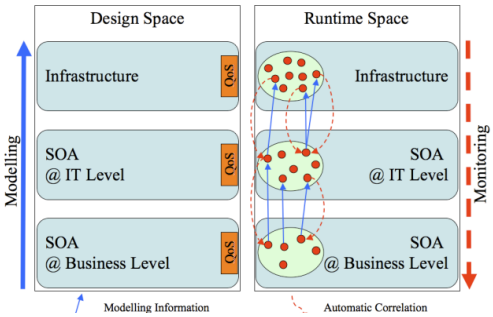


Figure 9 *Multi-layer perspective in SOA from [Mos et al. 2008]*

[Arsanjani 2004] descripts SOA as a partially layered architecture of composite services aligned with business processes. SOA provides a layer of abstraction that enables an organization to continue leveraging its investment in IT by wrapping existing assets as services that provides business functions. The relationship between services and support systems is that enterprise-scale support systems realize the services and are responsible for providing their functionality and maintaining their quality of service. Business process flows can be supported by a choreography of services that implement composite applications. An integration architecture supports the routing, mediation, and translation of these services, support systems, and flows using an Enterprise Service Bus (ESB). The deployed services must be monitored and managed to evaluate the quality of service and non-functional requirements.

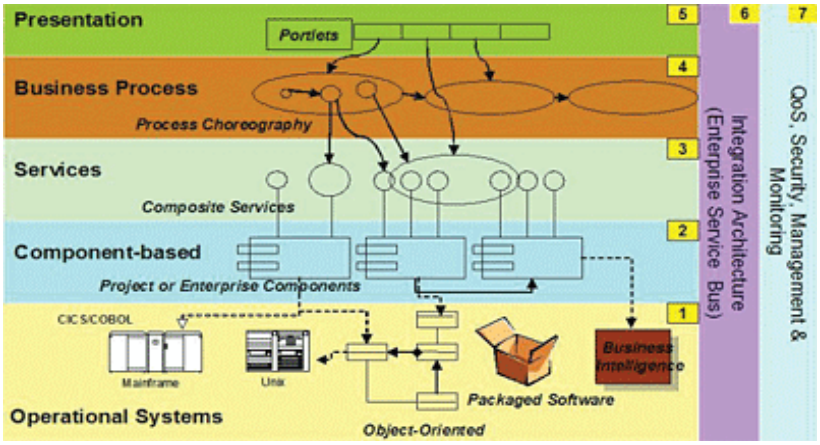


Figure 10 *High-level View of SOA Layers (from Business-driven development IBM: http://www.ibm.com/developerworks/webservices/library/ws-bdd/)*

From IBM's vision SOA integrates different layers:

Layer 1: Operational systems layer. This consists of existing customized applications, including existing CRM and ERP packaged

applications, and older object-oriented system implementations, as well as business intelligence applications.

Layer 2: Enterprise components layer. These enterprise components are responsible for realizing functionality and maintaining the QoS of the exposed services. As enterprise-scale assets, they are responsible for ensuring conformance to SLAs through the application of architectural best practices.

Layer 3: Services layer. It is a conceptual layer within a network service provider architecture. It aims at providing middleware that serves third-party value-added services and applications at a higher application layer. It also provides an interface to core networks at a lower resource layer.

Layer 4: Business process composition or choreography layer. Compositions and choreographies of services exposed in Layer 3 are defined in this layer. Services are bundled into a flow through orchestration or choreography, and thus act together as a single application.

Layer 5: Access or presentation layer. It is also important to note that SOA decouples the user interface from the components, and it provides an end-to-end solution from an access channel to a service or composition of services.

The ESB is not a layer but a transverse component. It enables the integration of services through the introduction of a reliable set of capabilities, such as intelligent routing, protocol mediation, and other transformation mechanisms, often described as the ESB (see Resources). Web Services Description Language (WSDL) specifies a binding, which implies a location where the service is provided. On the other hand, an ESB provides a location independent mechanism for integration.

QoS, Non Functional Requirements management and Non Functional Property Monitoring provide the capabilities required to monitor, manage, and maintain QoS such as security, performance, and availability. This is a background process aims to implement quality of service for a SOA.

[Papazoglou et al., 2007] extends SOA as xSOA. The architectural layers in the xSOA, embrace a multi-dimensional, separation of concerns in such a way that each layer defines a set of constructs, roles, and responsibilities. Each layer uses the lower layers- abilities to accomplish its mission.

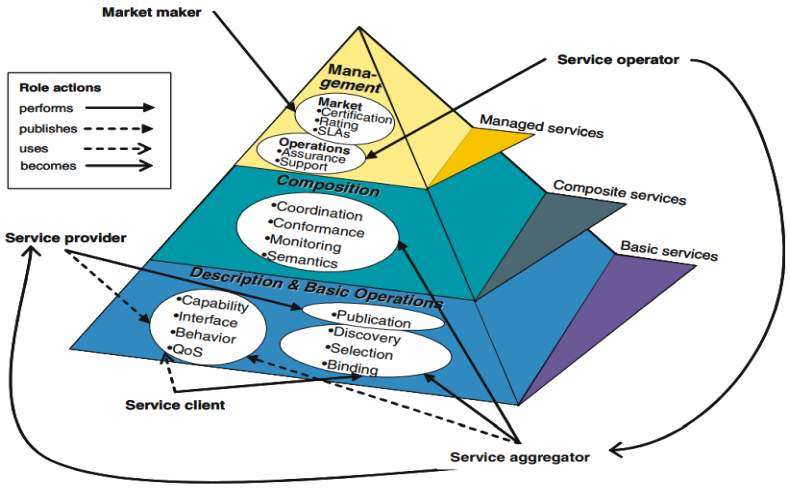


Figure 11 *Extended SOA (from [Papazoglou et al., 2007])*

As the extended SOA figure shows, the xSOA uses the basic SOA constructs as its foundational layer and adds service composition and management layer on the top.

- The Description & Basic Operation layer defines an interaction between software agents as an exchange of messages between service requesters (clients) and service providers.

- The service composition layer encompasses necessary roles and functionality for the aggregation of multiple services into a single composite service.

- The management layer defines service market management and service operations management. [Papazoglou et al., 2007] defines the most prominent functions of service management are Service Level Agreement management, auditing, monitoring and troubleshooting, dynamic service provisioning, service lifecycle/state management and scalability/extensibility. Service operations management is a critical function that can be used to monitor the correctness and overall functionality of aggregated/orchestrated services. It also should provide global visibility of running processes, comparable to that provided by BPM tools. Service market management's purpose is s to create opportunities for buyers and sellers to meet and conduct business electronically or aggregate service supply/demand by offering added value services and grouping buying power.

Enterprise Service Bus (ESB) is an infrastructure to facilitate SOA. Enterprise Service Buses (ESBs) are widely recognized as a mainstream middleware to support the service infrastructure layer of a SOA. The ESB is an open, standards-based message bus designed to enable the implementation,

deployment, and management of SOA-based solutions with a focus on assembling, deploying, and managing distributed SOA. ESBs provide a middle integration layer, with reusable integration and communication logic, which helps to address mismatches between services regarding communication protocols, message formats and QoS [González and Ruggia, 2010]. It is a way to integrate applications, coordinate resources, and manipulate information. The bus functions as both transport and transformation facilitator to allow distribution of these services over disparate systems and computing environments.

[González and Ruggia, 2010] characterizes ESB behavior through different patterns:

- Connectivity patterns include Service virtualization patterns; Service enablement patterns, Gateway patterns, Message-based integration patterns, File process patterns, Event-driven integration patterns.

- Mediation patterns include Intermediate routing patterns, Transformation patterns, and Management patterns which are designed to provide monitoring solutions to ESB infrastructure.

These basic patterns can be combined to fit complex situations. To successfully build and deploy a ESB, there are 11 functional requirements are listed by [Papazoglou et al., 2007]: Leveraging existing assets; Service communication capabilities; Dynamic connectivity capabilities; Topic/content-based routing capabilities; Integration capabilities; Transformation capabilities; Reliable messaging capabilities; Security capabilities; Long running process and transaction capabilities; Management and monitoring capabilities; Scalability capabilities.

Monitoring, service auditing and management are primary challenges to successfully build a distributed SOA. Industry analysts assume that a lack of SOA governance is the main reason why SOA projects fail. As an ESB connects different layers of SOA and transports messages to synchronize SOA's operations, it is a suitable place to execute monitoring tasks. Therefore, SOA Governance has evolved.

2.3.4 Non-Functional Properties and Service Level Agreement

Non-functional Properties (NFP) may refer either to the "business" vision associated to the service or to the software vision of the Service. According to [Aburub et al. 2007] Non Functional Properties might play an important role in all service related tasks, especially in discovery, selection and substitution of services. The requirements engineering community has classified requirements as either functional or non-functional. In software engineering, Functional Requirements (FRs) specify a system's functionality and behavioral, "what the

system should do, what the product must do". Although the term "Non-functional Requirement (NFR)" has no consensus over the years, different authors characterize the term NFR in different definitions, other authors use different name, as "Quality Requirement (QR)". Understand the performance level associated to NFP requires using a Performance Indicator system which should manage related indicators to measure the achievement of business resource objectives. Several kinds of indcators are commonly used to measure the performance of industrial organisations such as Supply chain (see for example [Cho, et al., 2012] [Giannis, et al., 2008] [Vlachos, 2014] that review the different indicators commonly used for supply chain performance evaluation). [Doumeingts, 1995] extends GRAI-GIM to design Performance Indicator System and to connect it to the decision organisation. ECOGRAI allows to specify measurement by splitting global objectives into detailed objectives before identifying relevant performance indicators. To fit the new requirement from integrated product-service value chain, beside industrial performance indicatorS, associated IT performance indicators should be ingrated into performance indicator system to achieve common organizational objectives. On the other hand, there is a unanimous consensus that NFRs affect different activities and roles related to the software development process, NFRs are critical for qualities of the system to be developed in software engineering field [Glinz, 2007], the functionality is not useful or usable when necessary NFRs do not hold [Chung, 2009]. [Rahman and Ripon, 2013] suggests that NFRs are one of the key criteria to derive a comparison among various software systems. However, identifying NFR is not an easy task. Although there are well developed techniques for eliciting functional requirement, there is a lack of elicitation mechanism for NFR and there is no proper consensus regarding NFR elicitation techniques. Eliciting NFRs are considered to be one of the challenging jobs in requirement analysis.

In industry area, the concept of NFR is borrowed from software engineering, and it is adapted and applied to business process modeling [Aburub et al. 2007]. NFR is a key competitive advantage to increase the chance of market success, it is important not only to develop a product that meets customers' requirements and expectation, but also to offer high value of products and services to increase customers' satisfaction. The importance of developing effective, efficient, quantified and testable methods for NFRs has attracted much attention in industry.

Although the term "NFR" has been in use for more than two decades, and the importance of NFR has been increasingly concerned in software engineering and industry practice. There are still open issues need to be resolved, such as elicitation and specification of NFRs, interdependencies among NFRs, impact of FRs and so on. This situation is particularly unsatisfying as today's

dynamic development of industrial product-service systems. There are several challenges as reported by [Loucopoulos et al. 2013], [Svensson et al. 2013]: difficult to elicit NFRs; often poorly understood; lack of consensus about NFR; generally stated informally in a non-quantifiable manner; difficult to be documented efficiently; difficulties to get attention for NFRs. Among these challenges, elicitation of NFRs attracted more attention than others, as reported by [Rahman and Ripon, 2013], [Zowghi and Coulin 2005] [Doerr et al. 2005]: elicitation is one of the crucial issues for the system development and a major part of the requirement engineering; lack of appropriate elicitation techniques for discovering NFRs; lack of appropriate solutions to refine NFRs with intertwined FRs and the architecture.

Due to the specific background and features of product service industry, product service industry not only faces the common challenges of NFRs with software engineering, but also requires a systematic support to deal with industrial specific challenges (see Table 4). The NFR Framework [Chung, 2009] provides a systematic treatment of NFRs in software engineering, but it does not provide any modeling notation to address the specific background and features of product service industry. Some of the referred features, which motivate our work to manage product service industrial NFRs, are: dynamicity, adaptability and customization; prioritization changes; context-awareness; service-oriented strategy; various measurements; etc.

Table 4 *Challenges of NFRs in software engineering and Industry*

Criteria	**Software Engineering**	**Industry**
Standard	Several case study show that a generic standard such as ISO-IEC 9125 is likely to be too general to be useful [Al-Kilidar et al., 2005] [Jung et al., 2004].	There is a possible mismatch between the established academic interpretation of quality characteristics of ISO/IEC 9126 and the industrial interpretation of NFR [Svensson, et al. 2009]. It requires a customized management for industry needs.
NFR Management in practice	Specification of NFRs, interdependencies among NFRs, and impact of FRs and so on, these are major challenges for NFR management in practice [Karlsson et al. 2007].	(1)Due to the complexity of real industrial context, it requires a wider survey to manage the specification of NFRs, independencies of NFRs, impact of FRs [Vara, et al. 2011]. (2)Due to the dynamicity of industry, the importance of NFRs is various. It requires a dynamic

		management to fit business needs [Svensson, et al. 2012]. (3)Due to the characteristics of industry, such as prioritization changes, context-awareness, heterogeneity of technological solutions, it requires some clustering to organize collected NFRs [Svensson et al. 2013].

As above mentioned, how to measure the NFR and how to deploy NFR is still a major problem of NFR [Rahman and Ripon, 2013]. Representation and elicitation are crucial challenges for NFR. With the development of emerging technologies and novel business paradigms, for industry both providers and consumers pay special attention on quality of products and services. This trend demands an increased attention to NFRs which deal with quality factors and satisfaction of customers. It requires an efficient way to manage NFRs, and this management should not only specify and cluster NFRs but also validate and guarantee the implementation of NFRs to bring benefits for industry practice. To sum up, a quality and NFR based governance method is required, and the objectives of this governance (such as, high quality, agile process, high customer satisfaction, etc.) fit the lean strategy which we have mentioned previously.

Non Functional Properties for business process can be identified into two dimensions. First, direct-service qualities represent qualities introduced directly to the customers. Second, indirect-service qualities represent general qualities that enable staff members perform their responsibilities efficiently and effectively. As such, it is relevant to integrate Non Functional Properties in tasks as services discovery, selection and substitution so that services that fit the functional requirements can be compared and ranked according to one or more Non Functional Properties (as cost, performance…). However, [Eenoo, et al. 2005] presents that due to various factors, we lack methodologies to support non-functional properties:

i. Non-functional properties are usually too abstract and most of the time they are stated informally;

ii. In most of the cases there is no clear delimitation between the functional and non-functional aspects of a service;

iii. Non-functional properties are often considered to be represented after the functional and behavioral have been described;

iv. Non-functional properties may often conflict and compete with each other (e.g. availability and performance);

v. Modeling non-functional properties is complex and difficult to formalize.

Despite the call for such a governance model involved by the recent economic turmoil and the practical implementation of Lean 6 six sigma principles into industrial organization. This lack of methodology could increase the difficulty of defining Business quality and monitoring it efficiently.

The rapid evolution of the telecommunications market is leading to the introduction of new services and new networking technologies in ever-shorter time scales. SLAs are tools that help support and encourage customers to use these new technologies and services as they provide a commitment from SPs (Service Providers) for specified performance levels [TM Forum 2008]. Quality attribute requirements play an important role in service selection in SOA environments. A SLA is part of the contract between the service consumer and service provider and formally defines the level of service. It defines the availability, reliability and performance quality of delivered telecommunication services and networks to ensure the right information gets to the right person in the right location at the right time, safely and securely. OASIS SOA reference model [OASIS 2008] suggests that Performance Metrics identify quantities that characterize the speed and quality of realizing the real world effects produced via the SOA service. In addition, policies and contracts may depend on nonperformance metrics. Some of these metrics reflect the underlying capability; some metrics reflect the implementation of the SOA service (see OASIS SOA Reference Model figure Figure 12).

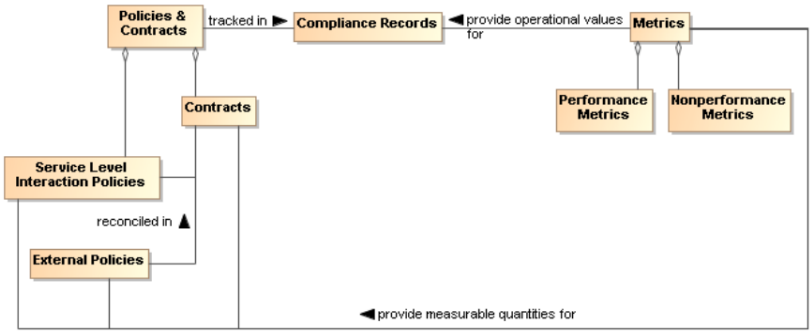


Figure 12 *OASIS SOA Reference Model Relating Policies and Contracts, Metrics and Compliance Records [OASIS 2008]*

Service Level Agreements are mandatory performance standards for given services. To honor the SLAs, services must be monitored with respect to information. [Marks and Bell, 2006] states that Producers of services should include these elements in their service contracts and SLAs:

- Consumption limits and ranges.
- Reusability and utilization parameters.

- Guaranteed service performance.
- State management methods.
- Quality assurance, Quality guarantees.
- Interface descriptions.
- Service availability.

Service consumers are charged by their consumption mean-while providers can be fined according to the violation according to elements in their SLAs.

From a Business perspective, a Business Agreement is an agreement entered into by two or more partners that constrains their future behaviors and permitted states. A Business Agreement is typically associated with business transactions. The transaction is guided by the agreement and an agreement can be the result of a transaction [OASIS 2008]. The goal of the enterprise SLA process is ultimately to improve the Quality of Experience (QoE) of the service or product to the enterprise clients, whether they are internal or external to the organization [SLA 2006]. A SLA is developed for these reasons [Bianco et al., 2008]:

- New commercial services entering a competitive market can use SLAs to establish themselves as reliable providers and hence attract customers.

- SLAs establish a commitment to quality levels required by service users and providers to interact effectively. Moreover, SLA management helps customers validate and supervise the quality of services through scheduled and on-exception reports.

- Many organizations provide services that depend on services from other organizations. In some situations, the unavailability or poor performance of one of the services may compromise the whole customer experience. In a multi-party composite service situation, SLAs can be used to identify the party that is responsible in case of a problem.

Service providers have created SLAs that specify key performance indicators (KPIs) associated with the communication infrastructure. More recently, the service management focuses on service quality rather than on network performance [TM Forum 2008]. SLA parameters are specified by a set of metrics. These metrics determine the measures that need to be gathered in order to verify whether the SLA parameters are being met or not.

However, SLAs do not pay much attention on the higher-level aspects of interaction between business and service-based applications. In a situation where a service-based application that has high availability and high responsiveness, delivers a low quality business activity, the business value of this service-based application will suffer. Therefore, the Business-Level Agreement (BLA) becomes a critical concern for enterprises to maintain the long-term business value of using these service-based applications. BLA is

complementary to the technical SLAs. BLA aims at making agreements on the quality of business activities realized through service-based applications. It defines measurement and metrics to monitor the service-based applications for conformance to BLA, allowing business value of service-based applications to be guaranteed. BLAs require a business framework within which to work. They additionally require an understanding of what impact business value can have in the application of a service. By understanding the measures and the most effective charging model, it is possible to free the enterprises from technical concerns and allow them to focus on operating their business effectively. BLAs for IT represent understanding the business service, the measures, the charging model and finally constructing a contract that motivates the partner in line with the business objectives.

Performing these agreements (SLAs /BLAs) requires SOA governance to define the set of policies, rules, and enforcement mechanisms for developing, using and adapting service-oriented systems, and analyzing their business value. SOA governance according to what their tools and products can measure and monitor. It requires these tools to define the measurement and definition of quality in SLAs. Nevertheless they do not cover other aspects of SOA governance, such as which services should be created, how they should be created, who should have access to them, and how will they be provisioned [Bianco 2008].

2.3.5 SOA Governance

SOA governance defines the organizational structures and processes that are required to successfully adopt SOA. It helps minimize complications. It also helps set the clear terminology and standards of communication crosses traditional enterprise boundaries and operational contexts. SOA governance increases the connection among business processes, functional capabilities, organizational entities, end users, applications and data.

SOA governance is strongly related to IT governance. Numerous IT governance frameworks have been specified, but each of them focuses on a different aspects of a company's IT, such as ITIL (IT Infrastructure Library), which mainly deals with IT process definition (Office of Governance Commerce: OCG), ValIT, ISO 20000, ISO 17799, etc. which targets security management (ISO) primarily. CObIT (Control Objectives for Information and related Technologies) by the IT Governance Institute (ITGI) is a governance and control framework, which is more closely aligned with the business objectives of the organization than with operational issues (IT Governance Institute) [ITGI 2007].

SOA governance begins with mapping corporate, business, and IT policies to identify specific SOA business services. Then it defines and enforces

the compliance rules and policies for managing those services, dictates policies for services reuse, IT compliance, and security. Strong organizational processes are critical to SOA success. The executive team must see SOA initiatives aligned with business goals, and strategies articulated with clearly defined project charters, requirements, and performance metrics.

Successful SOA governance occurs in several ways. SOA is often explored as a prototype at the business unit level to test the system internally and limit the investment risk. This "think big, start small" approach allows executives to measure results and prove value prior to implementation on a broader scale. SOA can also be implemented through incremental acquisition, where various business units are integrated in phases to manage the change. Either approach requires an internal champion to define, develop, and deploy SOA and demonstrate its benefits to the company months in any role before being moved to another role or leaving the company.

At the core of governance is the ability to monitor, measure, and analyze the organization's SOA service network. Business units must be managed at the micro-level, often using service level agreements (SLAs) to determine the performance benchmark. Policy enforcement is vital to ensuring that all business units work in tandem and use a standards-based process for interoperability.

SOA governance aims at disciplining an effective SOA. The basic idea is that SOA governance should support organizations to address any challenges from SOA implementation [Joachim et al. 2013]. Numerous models for SOA Governance have been proposed so far. All of them emphasize on different aspects, e.g., service lifecycle management (BEA Systems), organizational change, service integration testing [Bertolino and Polini, 2009]; describing models and tools for supporting SOA governance activities at the technical level [Derler and Weinreich, 2007], developing an SOA governance approach based on the lifecycle of single services [Schepers et al., 2008], or proposing new organizational structures for SOA [Bieberstein et al., 2005]. Some SOA governance models have been proposed by IT providers (as IBM, HP) who have defined SOA Governance within the context of business service lifecycles (SOA governance, IBM's SOA Foundation).

There are a lots of metric types used to complete the SOA Governance, such as business, process, performance, Service Level Agreement (SLA), and SOA conformance metrics [Marks and Bell, 2006]. Each of them corresponds to a specific type of policies. CObIT provides a set of common metrics. At the core of governance is the ability to monitor, measure, and analyze the organization's SOA service network. Business units must be managed at the micro-level, SLAs are often used as the services performance benchmark. SLAs have been a common product in support of services offered by telecommunications service

providers for many years. SLAs are now being considered for non-communications (network) services and are being adopted both internally and externally to define the agreed performance and quality of the service or product and as an important part of a Customer Relation-ship Management (CRM) program. The goal of the enterprise SLA process is ultimately to improve the Quality of Experience (QoE) of the service or product to the enterprise clients, whether they are internal or external to the organization [SLA 2006].

[Niemann et al., 2008] presents an approach for a generalized SOA Governance model. These authors identified six main components which form a mechanism for the optimal support of governance activities for an SOA system in a company. Most of the existing concepts, interaction schemes, and approaches to SOA Governance frameworks fit into their generalized SOA Governance model. One of the new key opportunities an SOA provides is the ability to cooper-ate with other companies more easily at a technical level. Nevertheless their SOA governance model, (as the most of SOA Governances) lacks of ability to govern the IT infrastructure and ignore the infrastructure performance which could impact the service performance.

[Bernhardt and Detlef, 2008] outline a reference model for SOA governance that is based on the standardized SOA-RM and motivated from aspects relevant to methodologies for SOA. Their model addresses governance aspects for the complete SOA lifecycle, stipulated in a set of governance policies, processes and organizational considerations, unlike previous approaches to SOA governance that are often limited to either design time or runtime aspects of SOA governance. However, their model is conceptual and they did not propose any approach to connect their model to common frameworks for IT governance and Enterprise Architecture. As a consequence SOA Governance methods, cannot give a comprehensive perspective of industrial governance to combine the IT infrastructure ability with the business benefits. To face the challenge of improving competitiveness we need to increase both enterprise and IT system agility and interoperability.

[Joachim et al., 2013] offers an evidence-based contribution to the discussion of the role of SOA governance when bringing together managerial and technical perspectives regarding service orientation. They conclude that SOA governance is crucial to reap the fruits sown through service orientation. Their analyses have shown the importance of SOA governance for SOA's ability to improve IT flexibility and services reusability. These findings complement the predominantly technical literature on SOA and also specify which governance mechanisms are needed to achieve increased integration, scalability, modularity, and reuse.

At a business layer, governance aims at managing business process, leading to Business Process Management (BPM) approaches. Business processes

need to adapt to changes in the operating conditions and to meet the service-level agreements (SLAs) with a minimum of resources. According to Toyota case study picked from Lean literature, business processes hide inefficiencies. One has to follow the flow of information as the design evolves into the finished product [Hammer, 1990].

The co-existence of SOA and BPM is used to support modern business needs [Jasmine 2005]. The joint venture of SOA and BPM is going to become a reality by converging SOA and BPM. Convergence of SOA and BPM is possible because the core functionality of the business processes in implemented through IT services. As a consequence IT services must be embedded somewhere in Business processes that are appropriate for both BPM and SOA. Colleen Frye [Bajwa et al. 2008] says that "BPM is a small fish inside the belly of the SOA whale…" In the same article, Colleen says also that "BPM and SOA are two sides of the same coin; joined at the hip".

To this end, we can see that SOA governance can ensure the combination of BPM and SOA to improve their business performance and align their business process with support system. In addition, it requires the SOA governance extending SLA to Business level, and providing the specific measurement and definition of quality.

2.3.6 Conclusion

The goal for a SOA is a worldwide mesh of collaborating loosely coupled services, which are published and available for invocation on the Service Bus. BPM and SOA are the counterparts in the modern business and information system's requirements. There can be many benefits of using the BPM and SOA in combination. Following are some advantages that can be attained by implementing both SOA and BPM in combination in a business enterprise [WLM 2007]:

i. The combination of BPM and SOA can reduce the cost of a business enterprise: operating cost, development and maintenance cost;

ii. Their combination can be helpful in speeding up the course of process creation and modification;

iii. Their partnership can also be used to increase the overall efficiency of a particular business enterprise;

iv. Complexity if the process model is decreased by enhancing the reusability factor;

v. The cooperation of BPM and SOA may support enterprises while they reorganize their Information System to set an agile and flexible system which can fit the dynamic changes context.

However, managing the interaction and independencies of loosely coupled services in SOA environment to fit the business objectives is still one of the main challenges to achieve benefits from combining BPM and SOA. These factors become even more complicated in a collaborative environment. In a collaborative environment services are delivered by different organizations within the company or even different companies (partners, suppliers, etc.). Delivering value to the stakeholders, auditing quality of service and performance of ISs, it requires a management and governance solution to manage these loosely coupled services, business processes and support ISs, and to govern their performance and quality. SOA Governance comes into the picture.

SOA Governance also meets some challenges:

(1). Change management: it requires consumer perspective governance, because of changing a service often has unforeseen consequences as the service consumers are unknown to the service providers. This makes an impact analysis for changing a service more difficult than usual.

(2) Ensuring quality of services: The flexibility of SOA to add new services requires extra attention for the quality of these services. This concerns both the quality of design and the quality of service. As services often call upon other services, one malfunctioning service can cause damage in many applications.

ISs as the support systems for implement of agile business, then the ultimate agility requirements come from the business layer. Business agility requires a better alignment of processes with business goals, making processes faster, more efficient, and more reliably compliant with policies and best practices, making business performance more visible even when the process crosses organizational or system boundaries, and more actionable in real time.

The benefits to be gained from combination of BPM, EA and SOA are listed as:

- Improvements in using IT to drive business adaptability.
- Alignment between business requirements and IT deliverables.
- Improve focus on organization goals.
- Reduced complexity of existing IT systems.
- Improved agility of IT and business.

All of these benefits are attractive for business participants to improve the performance of business value chain and business outcomes. Nevertheless in order to guarantee business gains these promised benefits, the quality of business processes and value chain should be closely monitored. A suitable performance measurement is required to implement SOA Governance to monitor the quality of business processes and business outcomes.

2.4 IT Agile Implementation Thanks to Cloud Computing

Aligning IT with business and achieving business agility are some of the most important challenges for enterprises to maintain their core competitiveness. EA framework and SOA are aiming at narrowing the gap between IT and business. The emerging Cloud Computing brings a new paradigm which is dramatically changing the way business support can be deployed. [Buyya et al. 2009] describes Cloud computing and emerging IT platforms as means to deliver computing as the 5th utility after water, electricity, gas, and telephony. In this section we introduce Cloud Computing, associated features and challenges related to our work. A number of computing researchers and practitioners have tried to define Cloud computing, after comparing with other distributed computing paradigms: such as cluster computing and grid computing. [Buyya et al. 2009] suggests that "A cloud is a type of parallel and distributed system consisting of a collection of inter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resource(s) based on service-level agreements established through negotiation between the service provider and consumers." The promise of cloud computing is to deliver all the functionality of existing information technology services (and in fact enable new functionalities that are hitherto infeasible) even as it dramatically reduces the upfront costs of computing that deter many organizations from deploying many cutting-edge IT services [Marston et al. 2011]. Cloud Computing and SOA have important overlapping concerns and common considerations, such as such as the focus on agility and flexibility. Both are based on services' reuse and producer/consumer model. Cloud computing and SOA are complementary and they work best together.

[Kim 2009] concluded four advantages offered by Cloud Computing:

- All computing resources and electricity needed for services are held by 3rd party. Service consumers only need to demand services by plugging into the cloud. There is no need of investment of computing resources, physical spaces and electricity, and the cost of maintaining all these resources.

- It is flexible for service consumers to increase or decrease required resources. It makes enterprises to scale their business flexibly.

- The pay per use economic model eases the cost control for consumers. It dramatically lowers the cost of entry for smaller firms being beneficial from cloud computing. So cloud computing represents a huge opportunity to business market.

- Service consumers can access to a large number of services from anywhere at any time. It increases the collaboration ability of business market, al-lows business to obtain benefits from technical easily.

Cloud computing represents a convergence of two major trends in information technology [Marston et al. 2011]: IT efficiency and Business agility.

The concept of IT efficiency does not only concern the computing resources used more efficiently. Futher, the computers can be physically located in geographical areas that have access to cheap electricity while their computing power can be accessed long distances away over the Internet. The concept of business agility does not just mean cheap computing, it also consider allow businesses to be able to use computational tools that can be deployed and scaled rapidly to meet ever-changing business needs.

2.4.1 Models of Cloud Computing

There are three core technologies will enable the evolution of Cloud Computing [Marston et al., 2011]: virtualization, multi-tenancy and web service. Virtualization is the technology that hides the physical characteristics of a computing platform from the users, instead presenting an abstract, emulated computing platform.

Everything as a Service (XaaS or EaaS) is a critical concept for cloud computing to implement its key enabling technologies (fast wide-area network; powerful, inexpensive server and high-performance virtualization). In the following we introduce the major types of services. XaaS targets to make the available resources consumable so that it could help businesses take advantage of cloud computing. Cloud-oriented Service Solutions could play an important part in transforming enterprise systems, contributing to cost reduction, agile deployment of services, expanded flexibility and improved productivity [Xu X. 2012]. The model of cloud computing relies on the delivery of different types of services.

- Software as a Service (SaaS): The software level is on the top of the infrastructure layer, with the services on this level providing on-demand applications over the Internet. Presently, there are many representative business products, such as mobile services provided by RIM Blackberry, Apple AppStore, Google Android Market and Location Based Services.;

- Platform as a Service (PaaS): The platform layer mainly refers to the software or storage framework which aims to minimize the burden involved with deploying applications directly into VM containers. Examples of PaaS include the Google AppEngine, Microsoft Azure and Amazon S3. In addition, some studies involving code migration also propose several code offloading architectures aimed to reduce the burden on application programmers [Young et al., 2001], [Cuervo et al., 2010];

- Infrastructure as a Service (IaaS): This layer creates a pool of storage and computing resources by partitioning the available physical resources by using virtualization technologies. The related commercial products of this layer include Amazon EC2, GoGrid and Flexiscale.

Other type of services can also be defined such as [Sultan 2013] defines business-related service type (Business Process as a Service – BPaaS; Management/Governance as Service – M/GaaS; Storage facilities related service type. [Zhang et al., 2010] describes a layered model of cloud computing. These authors divided a cloud computing environment into 4 layers: the hardware/datacenter layer, the infrastructure layer, the platform layer and the application layer:

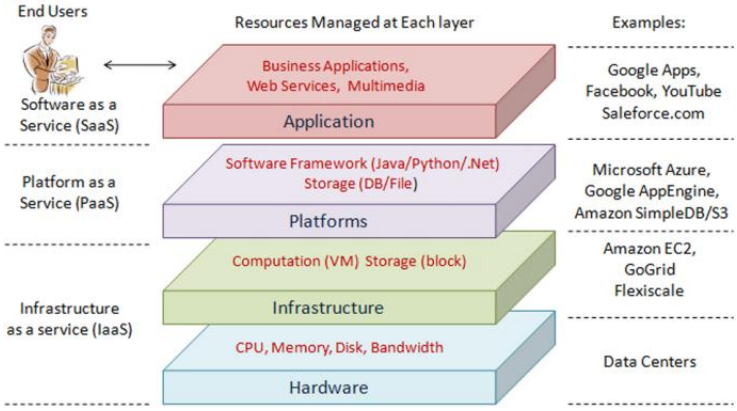


Figure 13 *Layered Cloud Computing [Zhang et al., 2010]*

Cloud Deployment Models including [Mell and Grance 2011]:

1. Private Cloud: The cloud infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on premise or off premise. This type of model is suitable for organizations focusing on privacy and data security, or to change or simplify the way people work. The downside is that implementations can be complicated, time-consuming or costly to complete.

2. Community Cloud: The cloud infrastructure is shared by several organizations and supports a specific community that has shared concerns. It may be managed by the organizations or a third party and may exist on premise or off premise.

3. Public Cloud: The cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services. This includes Cloud services offered in public domains such as Amazon EC2 and S3. This approach is for organizations wishing to save costs and time without obligations of deployment and maintenance.

4. Hybrid Cloud: The cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability.

2.4.2 Cloud Governance and Challenges

Even Cloud Computing has been changing the way we manage our business and computing resources are managed, it has become a growing interest for organizations looking to reduce their IT costs by offloading infrastructure and software costs onto 3rd party organizations who offer Everything-as-a-service. However, due to the relative infancy of this emerging cloud based computing, there are existing uncertainties obstruct utilization of cloud application. [Armbrust et al. 2010] identifies ten top obstacles for cloud computing:

1). Availability of a service: the risk of failure by "a single point of failure" and Distributed Denial of Service (DDoS).

2). Data Lock-In: the APIs for Cloud Computing itself are still essentially proprietary, or at least have not been the subject of active standardization. Thus, customers cannot easily extract their data and programs from one site to run on another.

3). Data Confidentiality and Auditability: Current cloud offerings are essentially public (rather than private) networks, exposing the system to more attacks.

4). Data Transfer Bottlenecks: Applications continue to become more data-intensive. Cloud users and cloud providers have to think about the implications of placement and traffic at every level of the system if they want to minimize costs.

5). Performance Unpredictability: sharing resources and the scheduling of virtual machines are inclined to problematic.

6). Scalable storage: because of short-term usage, no up-front cost, and infinite capacity on demand, it is difficult to apply persistent storage.

7). Bugs in Large-Scale Distributed Systems: it is difficult to remove errors in large scale distributed systems.

8). Flexible Pricing Model: some pricing model cannot response consumers' scaling requirement quickly.

9). Reputation Fate Sharing: Reputations do not virtualized well. One customer's bad behavior can affect the reputation of the cloud as a whole and it is difficult to transfer legal liability.

10). Software Licensing: it would increase its annual maintenance fee to at least 22% of the purchase price of the software.

[Marston et al. 2011] analyzed several impediments to cloud computing from business perspective. Many cloud applications do not yet have the availability or quality-of-service guarantees. Many enterprises hesitate to move to cloud because of the loss of physical control of the data that is put on the

cloud or because they cannot choose a reasonable pricing strategy for service provider and consumer to deliver economic value to business value chain.

Lack of governance in cloud computing is an obstacle for enterprises take advantage of new IT and business paradigm. Building an "integrative" and consistent cloud governance method could achieve cost transparency and reduction, service agility and quality, adopt cloud without disrupt but reinforces ongoing business processes. Such a flexible, efficient, low cost monitoring strategy could be the significant competitiveness of multi-layer architecture industrial organization, provided that performance related information could be attached to the convenient component, composed and monitored accordingly.

This increases the call for a consistent governance framework. A survey on Cloud monitoring [Aceto et al. 2013] suggests that Monitoring of Cloud is a task of paramount importance for both Providers and Consumers. On one side, it is a key tool for controlling and managing hardware and software infrastructures. On the other side, it provides information and Key Performance Indicators (KPIs) for both platforms and applications. [Aceto et al 2013] suggests that there are eight aspects that should be considered in Cloud monitoring: capacity and resource planning, capacity and resource management, data center management, SLA management, billing, troubleshooting, performance management and security management. Moreover, due to the complexity and scalability of Cloud Computing, it involves more complex monitoring systems are needed. These systems have to be robust, scalable and fast, to be able to manage and verify a large number of resources. [Aceto et al. 2013] also listed some open issues and future directions in Cloud Monitoring:

- Effectiveness: it requires custom algorithms and techniques; root cause analysis techniques and accurate measures.
- Efficiency: it requires algorithms and techniques are able to manage the large volume of monitoring data.
- New monitoring techniques and tools (cloud-ready);
- Cross-layer monitoring: it requires consider the comprehensive performance of resources which are functional separated to several layers.
- Cross-domain monitoring: it requires monitoring strategy is able to monitor multi-cloud (federated clouds, hybrid clouds, multi-tenancy services)
- Monitoring of novel network architectures based on Cloud.
- Workload generators for cloud scenarios: it requires workload balance strategy.
- Energy and cost efficiency monitoring: it requires minimizing the related energy consumption and cost.
- Standard and common test beds and practices: it requires the collaborative use of research facilities provides ways to share tools, lessons learned and best practices.

Besides, from different perspectives the Cloud monitoring can also be divided into two visions: Consumer-side monitoring vision and Provider-side monitoring vision. From the consumer side, the monitoring should focus on customer's interests and help consumer to understand the situation of the used services and optimize utility of services. From the provider side, the monitoring can be used to optimize provider's management system and give provider knowledge about internal functioning of different elements' performance and help provider to guarantee promised SLA and other service restrictions.

2.4.3 Conclusion

The emerging paradigm of Cloud Computing is changing the culture we deal with business and the way to invest IT. The XaaS is a critical concept in Cloud Computing. This concept makes enterprises can reconfigure their business resources and focus on their core competitiveness. The multi-deployment models allow fit various business situations. Cloud computing brings a number of advantages such as improvement of energy efficiency, optimization of hardware and software resources utilization, elasticity, performance isolation, flexibility and on-demand service schema. Cloud services are on-demand, elastic and scalable. From the business perspective, some features such as no up-front investment, lowering operating cost, highly scalable, easy access, reducing business risks and maintenance expenses make Cloud Computing attractive to business owners [Zhang et al., 2010],

However, building integrated ISs-business industrial organization in Cloud context is still limited due to some challenges that need to be addressed from business perspective, such as hiding complexity of information infrastructure from business perspective, maximizing resource utilization while minimizing energy consumption and consumer cost, guarantee the QoS and business performance, monitor resources timely state, data and process security, detect waste and risks along the cross-layer value chain and correct mistakes in time.

To overcome these challenges, we believe that a Cloud-ready Governance as a Service (GaaS) method could deliver flexible, high-efficiency, cost-effective and comprehensive governance to various multi-layer industrial architecture. GaaS aims to provide stable competitive power to industry enterprises and helps cloud computing systems and practices that support interoperability, portability and security requirements that are appropriate and achievable for various usage scenarios. However, due to the complexity of cloud infrastructure, it requires new performance measurements which can cross-layer to monitor all involved elements, it allows the monitoring results not only details but also should be comprehensive and understandable for consumers and

providers, as well as monitoring activities should energy and cost efficient. In the following we will introduce performance measurement and autonomic management to identify locks while adapting them to the Cloud Computing context.

2.5 Performance Measurement and Autonomic Management

2.5.1 Introduction

As stated in previous sections, building a SOA Governance strategy is a way to address existing challenges, such as guarantee QoS, manage SLA, and ensuring business performance, etc. The SOA governance requires adopting performance measurement and managing different performance indicators to fit the service-oriented environment. The role of these measures and metrics in the success of an organization cannot be overstated because they affect strategic, tactical and operational planning and control. Performance measurement and metrics have an important role to play in setting objectives, evaluating performance, and determining future courses of actions [Gunasekaran 2004]. Performance measurement metrics are specified into three levels: (1) strategy level, measures high-level organizational goals; (2) tactical level, deals with mid-level management decisions; (3) operational level, focuses on low-level operational objectives to implement tactical objectives. Integrating information system performance measurement into business performance measurement requires specifying and composing different performance metrics. Therefore, in this section we analyze existing performance measurement solutions with their advantages and existing challenges to adapt to SOA.

Firstly, we start with the brief introduction of the development of performance measurement. Then, we introduce some typical performance measurements' strengths and weaknesses, and analyze how existing performance measurement cannot fully fit the requirements in SOA. Lastly, we discuss how to adopt performance measurement to new paradigms.

Decades ago, the Performance Measurement (PM) aimed at representing processes thanks to measurements, analyses and responses. As time has passed, more complex performance measurement frameworks and systems have evolved [Folan and Browne, 2005]. In the 1980s the Japanese techniques and practices such as Lean and Six Sigma gave a competitive edge in global markets. These methods make extensive use of performance measurement to manage and improve performance of processes and organizations. However, at that time performance measurements only focused on financial information

[Banuelas et al., 2006]. The 1990s was a revolution period for performance measurement research. [Neely et al., 1995] defined performance measure as “the set of metrics used to quantify both the efficiency and effectiveness of actions”. Towards the 2000s there are several definitions for Performance Measurement System (PMS) with no consensus [Franco-Santos et al., 2007]. With the development of globalization, the increasing need for multi-culture collaboration increases PMS’ emphasis on servitization and the trend towards service-dominant logic [Chesbrough and Garman 2009].

As a result of globalization, collaborative environments are increasingly required to be autonomous, scalable, adaptive (to fit dynamic changes in the net-work), and to exhibit survivable capacity against partial system failures, waste and cause of defect. In such environment, business decision-makers face a large-scale network application such as data center applications and grid computing applications have been increasing in complexity and scale. They are increasingly required to address critical challenges such as autonomy (the ability to operate with minimal human intervention), scalability (the ability to scale to a large number of net-work hosts and users), adaptability (the ability to adapt to dynamic changes in network conditions (e.g., network traffic and resource availability)), and survivability (the ability to retain operation and performance despite partial system failures (e.g., network host/link failures)) [Vasilakos et al. 2008].

Moreover, the integrating several heterogeneous environments into corporate-wide computing systems and extending that beyond company boundaries into the Internet, introduces new levels of complexity. Collaborative environments [Fawcett S.E. et al. 2008] have to face efficiently changing to meet the requirements of environment, being self-adaptive, self-organized, robust and allowing distributed and parallel computation as well as self-learning). As systems become more interconnected and diverse, architects are less able to anticipate and design interactions among components, leaving such issues to be dealt with at runtime. Soon systems will become too massive and complex for even the most skilled system integrators to install, configure, optimize, maintain, and merge. There will be no way to make timely, decisive responses to the rapid stream of changing and conflicting demands.

This leads to new approaches to manage complexity to overcome these monitoring challenges of monitoring and constraint for business in cloud environment, autonomic computing propose a new approach for computing systems that can manage themselves given high-level objectives from administrators [IBM 2011].

2.5.2 Performance Measurement Analysis

[Bourne et al. 2000] lists three stages for the lifecycle of Performance Measurement system: 1) Designing PMS, 2) Implementing PMS, 3) Using and updating PMS. Besides, design a PMS, the Implementation and Updating PMS also need more attentions. Despite of that, most of related PMS focus on financial performance. [Van Der Stede, et al., 2006] points out that integrating non-financial measures will achieve higher performance. An integrated comprehensive PMS can fully measure various aspects an enterprise performance. However, performance measurement could fail due to complexity of measurements. For Using and Updating PMS, in this ever-changing business environment enterprises depend more and more on collaboration and have to share performance information. However, performance behaviors depend on different management strategies [Aedo et al. 2010]. As a consequence the relevant performance indicators and measures should be reviewed to sustain their relevance with business management strategies [Bourne et al 2000].

[Bourne et al 2000] and [Kennerly and Neely 2003] pointed out that implementation PMS consists in four tasks: 1) data creation, 2) data collection, 3) data analysis and 4) information distribution. Management Information System (MIS) is playing an important role in today's business environment. A number of research has established links between PMS and MIS. IT development can leads to enrich PMS with new functionalities whereas MIS is required to support decision making by delivering data collection, analysis and storage [Haag et al. 2002]. This knowledge improves efficiencies of business operations [Marchand et al. 2000] and communications. The combination of PMS and MIS research model to set Performance Management Information Systems is suggested by [Marchand and Raymond 2008]. Performance information practices are required to convert data from internal and external sources into performance information before communicating these performance information, in an appropriate form, to managers at all levels in all functions to enable them to make timely and effective decisions [Bourne et al. 2000]. [Tsakonas and Paptheodorou 2008] suggests that open access to information will be beneficial to the business utility. Performance information measurement is not only technical strategy also involving people's behavior, [Kraaijenbrink 2007] concluded that there is wide gap in people identifying, acquiring and using the information. [Franco-Santos et al. 2007] suggest that PMS should encompass rewarding or compensating behavior, managing and control people's performance measurement behaviors. All in all, PMS is a prominent approach to benefit organizations by successfully implementing PMS through success factors such as business management commitment, people's appropriate behavior and MIS support.

In order to proactively respond to these challenges, management requires up-to-date and accurate performance information on its business. Performance information needs to be integrated, dynamic, accessible and visible to aid fast decision-making to promote a proactive management style leading to agility and responsiveness. Despite the amount of research and development in PM, there are only few systems that are properly integrated, dynamic, accurate, accessible and visible to facilitate responsive manufacturing and services. The existing PMSs have some common problems [Nudurupati, et al., 2011]: most PMSs are historical and static and only few PMSs have an integrated Management Information Systems (MIS) infrastructure, these systems lack of flexibility to fit ever-changing business requirements.

[Neely et al., 2000] has complained that much of the current research on PM, and the related literature about PMSs and frameworks are too superficial. There are very few PM systems in existence that have been academically developed. We take these three academic PM systems as being representative of the available PM systems' literature (see Table 5)**.**

Table 5 *Comparison of three PMs*

PM Solution	Contribution	Problem
1. [Kaplan and Norton, 2001] proposed a balance scorecard based PM system:	- It consists of an extended PMS approach and PM framework focusing upon objectives, measures, targets and initiatives. - Dimensions of measurement include: Financial, internal business, customer perspective, innovation and learning.	- No specific measures. Even in their additional procedural framework, performance measures are not explicitly pre-defined by the approach, which relies upon the system design methodology to formulate them during the system building process. It took a long time and consumed a lot of manpower to develop a balanced scorecard.
2. [Bradley 1996] proposes a performance measurement approach to the reengineering of manufacturing enterprises	- It specifically addressed Business Process Reengineering processes and included items that were new to performance measurement at the time. - Pre-defined lists help to reduce the amount of subjectivity required of	- Lack of dynamic. It may result in a certain loss of flexibility of the methodology. Also predefined lists are not adaptable for updating and are not easily meet other requirements.

	the PM system process - Dimensions: business processes, competitive priority, manufacture environment.	
3. [Medori and Steeple, 2000] proposes a framework for auditing and enhancing performance measurement systems	- It embraces both the design and auditing of PM systems. - It also introduces a specially-designed procedural framework for PM system design; they are effectively detailing the components of a system. - Dimensions: quality, cost, flexibility, time, delivery and future growth.	- Lack of dynamic. There are two problems identified with the system are: difficulties can be found in relating a company's strategy to the performance measurement grid's competitive priorities; and the separate pre-defined list of performance measures may become dated.

The first significant extended enterprise PM framework is in the work of [Brewer and Speh, 2001]. This framework moves beyond more traditional supply chain PM frameworks by expanding the concept of the internal perspective of the scorecard to include inter-functional and partnership perspectives. [Lohman et al., 2004] have pointed out various barriers to designing and implementing supply chain wide PM systems due to the allowing limitation factors:

- Decentralized, operational reporting history;
- Deficient insight in cohesion between metrics;
- Uncertainty about what to measure;
- Poor communication between reporters and users;
- Dispersed information technology infrastructure.

Performance measurement focuses on results, it aims to discover innovative ways to increase profits, reduce costs, predict trends and turn information assets into true competitive advantage. Despite the amount of research and development in performance measurement, the technical and people issues concerning the dynamics of performance management systems are

not completely understood [Folan and Browne, 2005]. To this end, it motivates our research to propose a combination of PMSs and Management Information Systems (MIS) with strategy maps, dashboards and financial statements. Strengthening the value of PMSs, Dashboard provides an efficient way to manage performance measurement results.

2.5.3 Dashboards Management

A dashboard is an easy to read way to present Performance Measurement results. Such dashboards are often presented as a single page, real-time user interface. It shows a graphical presentation of the current status (snapshot) and historical trends of an organization's Key Performance Indicators. This enables instantaneous and informed decisions to be made at a glance in management information systems [McFadden 2012]. [Negash and Gray 2008] considers dashboards as one of the most useful analysis tools in Business Intelligent. [Yigitbasioglu and Velcu 2012b] suggests that in more recent years, dashboards have evolved from the intrinsic purpose of monitoring performance to more advanced analytical purposes, incorporating new features such as (i) scenario analysis, (ii) drill down capabilities, (iii) and presentation format flexibility (e.g. tables or graphs). This claim is supported by the success stories reported in [Yigitbasioglu and Velcu 2012a].

[Pauwels et al. 2009] suggests four possible purposes of using dashboards:

(i) Monitoring refers to the day to day evaluation of metrics that should result in corrective action.
(ii) Consistency relates to the alignment of measures and measurement procedures used across departments and business units.
(iii) Planning gives scenario analysis with present features.
(iv) Communication, a dashboard communicates both performance and the values of an organization to its stakeholders through the choice of the metrics

[Wiersma 2009] identifies three purposes regarding the use of the balance scorecard, which may be applicable to the dashboards context: (i) decision-making and decision-rationalizing; (ii) communication and consistency, and (iii) self-monitoring. [Eckerson 2005] identifies three purposes: (i) Monitoring: convey performance status and trends at a glance; (ii) Analysis: analyze exceptions and find root cause; (iii) Collaboration: collaborate, plan and act.

Dashboards and scorecards are both visual interfaces for monitoring business performance. They present the status of Key Performance Indicators

(KPIs) monitoring information. However, what's the difference between a dashboard and a scorecard? To consider this we compare dashboards with scorecards from following aspects (see Table 6):

Table 6 *Comparison between Dashboards and Scorecards*

	Dashboards	**Scorecards**
Purpose	Measure current activity	Summarize progress
Updates	Run time	Periodic snapshots
Display	Loosely defined charts	Defined symbols
Focus	Monitor operations	Execute strategy
Scope	Operational	Enterprise
Information	Detailed	Summary

In a summary, a scorecard measures performance against goals, typically a scorecard is based on a collection of Key Performance Indicators which provide a snapshot of organization performance in a periodic time point. A dashboard is a container for various types of presentation. A typical dashboard might contain a scorecard, an analytic report, and an analytic chart. It provides a clear picture of current organization's operational performance. As a consequence, for a runtime detailed monitoring a Dashboard is more convenient than a Scorecard. While for an enterprise level periodic summarize a Scorecard is more convenient to present organizational strategy status.

Due to these requirements combining the advantages of dashboards and scorecards and providing customizable presentation appear as a convincing way to cope with our problems. Nevertheless, we identify few challenges to overcome: (i) Selecting Key Performance Indicators to measure and indicate the most important aspect of business performance. (ii) Well designed dashboard will give a clear and customized presentation, chose right charts and convenient category. (iii) Showing numerical information in an understandable way from business perspective and analyze data for further business decisions.

Due to the complexity of organization's collaboration, to monitor business performance precisely, a large amount of Key Performance Indicators is required. Therefore, it requires an effective and efficient management strategy to manage Key Performance Indicators and monitoring processes. A strategic view is necessary to manage the massive Key Performance Indicators in the complicated context dynamically and allow IT focus on providing higher-value outcomes which can improve business performance. IBM introduces the Autonomic Computing is a solution to make IT smarter and fit business rules to build a self-configuring, self-healing, self-optimizing and self-protecting system.

2.5.4 Autonomic Management/ Autonomic Computing

Managing efficiently complex information systems requires efficient means to deal with changes rapidly, to understand high-level objectives and to be able to adjust detail operations to achieve global objectives. Autonomic Computing, which is inspired by biological systems, provides a way to deal with these challenges.

The essence of autonomic computing systems is self-management. Its intent is to free system administrators from the details of system operation and maintenance and to provide users consistent high performance. [Sterritt, 2005] identifies eight elements of Autonomic Management: (1) Detailed knowledge of system components. (2) Self-configure and reconfigure. (3) Optimise operations. (4) Recover without harm system processes. (5) Self-protection. (6) Allow be aware of environment and adapt. (7) Function in a heterogeneous context. (8) Hide complexity. Like their biological namesakes, autonomic systems will maintain and adjust their operation in the face of changing components, workloads, demands, and external conditions and should face both innocent and malicious of hardware or software failures. The autonomic system might continually monitor its own use, and check for component upgrades. When it detects errors, the system will revert to the older version while its automatic problem determination algorithms will try to isolate the source of the error.

Due the problems found in a collaborative environment are quite similar to those encountered in a biological system. Both systems have to keep stable in a changing environment, should have efficient adaptability to face the ever-changing situation, such as diversity, self-tolerance, distributed and parallel computation, self-organization, self-learning, self-adaptation, and robustness. Achieving autonomic management in ISs supported business context, fits the loosely coupled Cloud context, key ideas for autonomic systems are inspired by biological immunology.

The primary function of a biological immune system is to protect the body from foreign molecules known as antigens. It has great pattern recognition capability that may be used to distinguish between foreign cells entering the body (non-self or antigen) and the body cells (self). Immune systems have many characteristics such as uniqueness, autonomous, recognition of foreigners, distributed detection, and noise tolerance [Castro and Zuben, 2002].

Our natural immune system protects our body from foreign cells called antigens by recognizing and eliminating them. Our immune system constitutes a self-defense mechanism of the body by means of innate and adaptive immune responses. An adaptive immune response contains metaphors like pattern recognition, memory, and novelty detection. The fundamental components of the immune systems are lymphocytes or white blood cells, which are divided into

two classes: B- and T-cells. B-cells have a more important function than T-cells be-cause each B-cell has its distinct chemical structure and secretes many antibodies from its surface to eliminate the antigens. A huge variety of antibodies are generated to neutralize and eliminate antigens. Each antibody is constituted by a specific B-cell whose aim is to recognize and bind to antigens. In the immune system, an important function is clonal selection. Clonal selection determines how an immune response is given when an antigen is detected by a B-cell [Aydin et al., 2010].

Inspired by biological immune systems, Artificial Immune Systems (AISs) have emerged during the last decade. They are incited by many researchers to design and build immune-based models for a variety of application domains. Such as in detecting anomaly network hosts [Gonzalez and Dasgupta 2003], in peer-to-peer content discovery [Ganguly and Deutsch, 2004], in product recommendation [Chen and Aickelin 2004], etc. The negative selection process is applied in [Gonzalez and Dasgupta, 2003]. Immunologically inspired strategies have been successfully used in network security [Saiz, et al., 2005] to detect incursions. [Basu 2012] presents artificial immune system for combined heat and power economic dispatch. [Chen 2010] presents an agent-based artificial immune system approach for adaptive damage detection in distributed monitoring networks. These approaches establish a new monitoring paradigm by embodying desirable immune attributes, such as adaptation, immune pattern recognition, and self-organization, into monitoring networks. [Venkatesan et al., 2013] proposes a platform protection mechanism by incorporating the artificial immune system. Apart from above we have mentioned application, artificial immune system have been applied for many models like anomaly detection including intrusion detection [Boukerche et al., 2007][Tarakanov 2008], computer security [Harmer et al., 2002], and misbehavior detection [Dasgupta et al., 2005] because of its efficiencies.

2.5.5 Conclusion

As existing Performance Measurement Systems are designed for fix context, they cannot directly be applied to dynamic and loosely coupled Cloud context. Implementing a Cloud-ready performance measurement leads to address some challenges: customizing metrics, configuring measurement parameters dynamically, analyzing measurement results for business needs pay attention on business process organization, computing loosely coupled results to reflect global business performance, triggering performance optimizing actions to improve business performance.

In order to overcome the above challenges on performance measurement in cloud environment, a standard approach combining techniques

and people behaviors to strength business competitiveness is required. Moreover, monitoring a massive number of virtual resources, managing a large number of Key Performance Indicators efficiently and allowing ISs to deliver maximum value for business with minimum cost, require also an autonomic management for the performance measurement. Implementing autonomic management, self-management is the main challenge.

As we have discussed immunologically inspired approaches have their advantages to achieve autonomic management. They have been used in different complex fields because of its efficiencies. We apply the artificial immunity theory to build Dynamic Evolution of Self set Model, Dynamic Immune Tolerance Model, Mature Key Performance Indicator Lifetime Model and Dynamic Memory Key Performance Indicator Model to automatically control governance detectors, to detect system failures, unexpected activities, waste and cause of defects, as well as trigger action engines to resolve problems which have been detected timely. Satisfying the ever-changing requirements the dynamic self-adjustment is necessary for governing business performance. The dynamic self-tolerance and clone selection algorithm could improve the agility of self/non-self's definition. We can add new self-elements into, or eliminate the mutated ones from self-set, resulting in the dynamic evolution of self set, mature and memory detectors, to reduce governance error rate.

Although AIS models have achieved great successes in various application domains, there are still some theoretical issues that need to be further explored such as the development of unified frameworks, convergence, and scalability. The developments of the artificial immune systems would benefit not only from the inspiration of biological immune principles and mechanisms, but also hybridization with other soft computing paradigms, such as neural net-works, fuzzy logic, and genetic algorithms. They could also be further studied and applied to more challenging application areas and to solve complex real world problems.

2.6 Conclusion

Developments in the global economy have amplified the need to consider not only productivity but also how to address customer needs more astutely and how to capture value from providing new products wish valuable services [Teece 2010]. Due to the collaboration and globalization, information plays more and more important role in business context. Information value flow supports and adds value to global business value chain. Achieving business agility, the quality and agility of information value chain is a critical success factor. Applying Lean thinking to information systems is a way to improve the quality and agility of

information value chain. Due to the ultimate agility requirement from business level, information value flow as a support factor should be organized match the business goals. However, making information value flow fit the requirements of business value flow is a main challenge to achieve Lean information value chain.

Combining Business Process Management (BPM) and Enterprise Architecture (EA) is a way to implement complex IT solutions and align the IT with business model. However, EAs do not provide a clear guidance to measure and govern whether the established EAs fit the goal of BPM or not. On the other hand BPM lacks the architectural principles, policies, and standards to integrate with EA. Enhancing the interaction and communication between BPM and EA is still a main challenge.

SOA provides a Service-Oriented way to organize enterprise architecture and business resources. It aims at narrowing the gap between IT and business. However, organizing a large number of loosely coupled services to satisfy business needs and paying attention on Quality of Service (QoS) to improve business performance are still critical challenges.

The emerging paradigm of Cloud Computing with its Everything as a Service (XaaS) concept strengths the advantage of Service-Oriented solution. However, due to the complexity of Cloud environment, there are still some challenges that need to be addressed: such as availability, concurrency and dynamic reconfiguration. From business perspective the most concerns include maximizing resource utilization while minimizing energy consumption, ease consumer cost, guarantee the QoS and service performance, monitor resources timely state, data and process security, detect risks and correct mistakes in time.

This trend leads to a new Service-Oriented industrial organization solution, which should adjust ISs to enrich business value, achieving business objectives, maintain business competitiveness and allow self-adjustment to fit new business environment. Achieving this Service-Oriented industrial organization and ensuring enterprise's IT investment fits business goals, it requires performance measurement for information value flow and business value flow.

Due to the changes of information technology and business models, the fixed Performance Measurement Systems (PMSs) are not fit the distributed loosed-couple environment. Difficulties of measuring massive resources, lack of further actions to correct performance, lack of customized measurement settings are main challenges for adapting PMS to Cloud environment. Managing a large number of Key Performance Indicators (KPIs) and performance optimizing actions requires autonomic management. Implementing Self-management is the main challenges of achieving autonomic management.

To this end, it requires a novel performance measurement solution with autonomic management strategy. This novel measurement should measure the

performance of information value chain and business value chain cross layers, cross domains. It should also custom algorithms with autonomic management and it should cover the multi-level agreement management. The complexity of governing massive resources and managing a large number of Key Performance Indicators (KPIs) in Cloud context is still a challenge to achieve a dynamic performance management solution.

Part 2 A Multi-layer Service-Oriented Management and Governance Architecture

3 A Multi-layer Service-Oriented Management and Governance Architecture

3.1 Introduction

Due to the market evolution (globalization, mass customization, sustainability requirements…) and emerging business paradigms, enterprises not only focus on productivity but also pay attention on improving customer satisfaction and adding more value to business outcomes. Enterprise Modelling methodologies and Enterprise Architectures are developed to organize and integrate different components of enterprise to achieve global objectives. However, these methodologies lack of quality assurance and flexibility. Moreover, with the development of information technology, Information System (IS) becomes a critical factor to achieve business objectives. Adding value picked from information system to business value chain requires to integrate information system with business organization, paying attention on IS artifact associated to real industrial resources. Due to the importance of information systems, the quality of information value chain is a critical success factor for implementing business value chain agility. Applying Lean thinking to information systems is a solution to improve the quality and agility of information flow. However, integrated information technical value to business value flow, identifying information value from business perspective, organizing information value stream to fit business needs are main challenges for applying Lean thinking to information value chain and ultimately achieving business agility. It requires a new value flow management which should pay attention on both information value chain and business value chain to realize the integration of these value chains.

Integrating information value flow into global business value flow, combing Business Process Management (BPM) and Enterprise Architecture (EA) can help organizations implementing complex IT solutions and aligning IT with business model. However, EAs lack a success measurement and a governance guidance to build a clear and success integration between IT and business model. On the other hand, BPM lacks the architecture principles, policies and standards to link business strategies with IT. These barriers impede the integration of information value flow into business value chain. It requires a new enterprise architecture solution which can help organizations to measure the IT

investigations benefit business outcomes and the success of EA can be tested from business perspective.

SOA provides a loosely coupled distribute solution to narrow the gap between IT and business. Its reused service-oriented concept aims at improving the agility and flexibility thus making organizations change to new market needs more cost effectively. However, managing the quality of a large number of loosely coupled services efficiently and ensuring the orchestration of services fit business goals are main challenges for organizations obtain the promised benefits from SOA.

The emerging paradigm of Cloud Computing provides ease of access to and usage of services with low cost which changes the way we are doing business. However, due to the complexity of Cloud Computing, there are still some challenges need to be addressed for making organizations success in Cloud context. Such as the virtualization brings difficulties of identifying real information value to support business value flow for cross-domain participants, thus it is difficult to architecture all of distributed resources along organizations' business value proposition. Due to the diversity of service and the changings of consumers' requirements, it is difficult to control the Quality of Service (QoS) and to manage the performance of cross-layer business value chain to meet organizations' ultimate business goals.

To govern and manage services in SOA and Cloud context, a cloud-ready Performance Measurement System (PMS) is required. It should also couple information quality performance indicator with industrial business performance indicator to get a global performance vision. However, existing PMSs cannot fit the Service-Oriented requirements. They are rather fixed and difficult to measure massive resources. Moreover, Non Functional Requirements (NFRs) are important factors deal with quality of industrial business solution and satisfaction of customers, the ultimate business agility requirements are from business level's NFRs. Transforming the business agility requirements to IT level and specifying the requirement to information value stream are still major challenges need to be addressed. The importance of developing effective, efficient, quantified and testable product service system requires an efficient way to manage Non Functional Properties. However, measurement, representation, elicitation and quality improvement of Non Functional Properties are still major problems for Non Functional Properties managements. In addition, lack of further actions to improve quality and performance of measured objects is still a limit to implement performance measurement in Cloud context. Achieving performance measurement in Cloud context, it also requires autonomic management to manage a large number of Key Performance Indicators (KPIs) and to provide effective and efficient measurement with low cost.

To overcome these existing limits, we propose a Service Oriented industrial organization that should narrow the gap between business and IT in Cloud context and improve the business agility by paying attention on ISs agility. To achieve this Industrial Service Oriented Organization, we build a multi-layer governance architecture, which consists of a top Business Decision Layer, three horizontal Business Resource management layers (BPL: Business Process Layer; BSL: Business Service Layer, BIL: Business Implementation Layer) and a vertical Governance Execution Layer. Business Resources are defined as all business process implementation elements including Business Tasks, Services, Infrastructure elements (such as hardware, equipement, database, software, etc.). The horizontal layers (BPL, BSL and BIL) aim at deploying all Business Resources and compositing business objects with IT objects.

To improve agility and quality of business, to improve ability of enterprises to cope with changes from both a technical and a business point of view, as well as to reinforce external and internal collaborative work of enterprises, we should make sure all business related activities and resources are organized efficiently and effectively, and evaluate business processes to guarantee they do add value to end-users without waste and defect. Aligning business with IT and narrowing the gap between different layers in Cloud environment (platform as a service-PaaS, software as a service-SaaS, and infrastructure as a service-IaaS) is a key goal for an efficient deployment of a lean strategy. We enrich the Everything as a Service (XaaS) traditional services specification with an agile Business as a Service (BaaS) level. This level aims at managing and integrating different components of enterprises to achieve a global business objective. Furthermore, we pay attention to the performance quality of essential elements “activities” and “resources” in this BaaS layer.

To this end, we propose a three-phase Governance Loop and two frameworks to implement our Service Oriented Management and Governance Architecture (SO-MGA). The Governance Loop includes a Governance preparation Phase which has to identify the governance objects in a specific context, a Governance Execution Phase which is in charge of deploying governance elements to implement automatic governance processes, a Governance Adaption Phase used to adjust governance elements’ parameters to fit dynamic environment and improve quality of governance objects automatically. To support this governance loop, we have defined two frameworks. The Business as a Service Management and Governance Preparation Framework (detailed in chapter 4) aims at managing and optimizing resources of BaaS for business in multi-cloud environment. It also prepares governance required information for the subsequent governance processes. Then the Business as a Service Governance Execution and Adapting Framework aims at taking customized requirement to monitor resources BaaS at runtime

dynamically composes initial real-time results to comprehensive results, and may improve performance of BaaS automatically according to these governance results.

In this chapter we introduce the working principles and main elements of these two frameworks generally. Besides, to simplify the complexity of message exchange and internal cooperation in this multi-layer SO-MGA, and to provide unified interaction format, we also introduce our proposed Integrated Management and Governance Bus and a general definition of resources in BaaS and the classification of Non Functional Properties.

The rest of this chapter is organized as following: we start with the governance loop, structure and layers of this SO-MGA (section 3.2). We pay attention on the general definition property of Resource (section 3.3) and generally introduce the organization of Patterns and Governance Rules in our multi-layer architecture (section 3.4). After that we present the major components of this SO-MGA, the working principle of these components (section 3.5). This chapter ends up with a chapter conclusion (section 3.6).

3.2 Service-Oriented Management and Governance Architecture

As we aim at integrating both industrial and IT vision, we propose to extend the commonly adopted Cloud service model (SaaS: Software as a Service, PaaS: Platform as a Service, IaaS: Infrastructure as a Service), to a Business as a Service level. This level aims at coordinating different Business Resources (including business and IT objects that are IT artifacts associated to real industrial resources) to achieve organization objectives. We define Business as a Service (BaaS) as an integrated business solution including functional (tangible product or a service) and non-functional properties. Functional properties (FPs) define “what jobs have to do” and Non Functional Properties (NFPs) defines “how these jobs are implemented”. These two families of properties are complementary. Non Functional Properties not only increase value of the sold product or service but also constrain performance of them. In a symmetric way Non Functional Property own “quality” may depend on the “functional side”, i.e. the sold product or service can impact the quality of Non Functional Properties. Non Functional Properties are critical success factors for managing and governing quality of BaaS. All of these properties are taken as governance resources. Organizing quality governance should pay attention on Non Functional Properties and consider the independencies of Functional Properties. In addition, in Cloud context applying Lean thinking to improve quality of information value flow the integration of cross-layer technical value and global business value chain is required.

To this end we propose a Service Oriented Management and Governance Architecture (SO-MGA). This SO-MGA aims at:

1) Providing and managing BaaS in multi-cloud environment: This includes an efficient organization of business resources to fit Business Decision Maker (BDM)'s requirements, increasing the resources ability. It has also to manage negotiation and agreement. Providing a dynamic BaaS organization and paying attention on QoS and Non Functional Properties.

2) Narrowing the gap between business request and technology implementation: In order to allow Business Decision Makers to efficiently govern and improve the quality of business value chain from business level to infrastructure level. This includes avoiding any monitoring blind spot in Business Process and narrowing the gap between business, service and infrastructure. It also aims at increasing both enterprise and information system agility and interoperability; reducing wastes and errors in Business Process and enhancing the robustness of business in multi-cloud environment; maximizing the usage of resources and commercial value, etc. In complex business context it requires to compose related individual monitoring results to comprehensive result (see). To get a global view of governing Business as a Service in product service industrial context see Figure 14.

In order to achieve such customized governance objectives, the governance environment should be analyzed to extract the critical success factors. These critical success factors are concluded as our governance objects. Governance parameters must be adjusted according to business needs to fit specific situations. To this end, we propose a governance loop which includes following three main phases: Preparation, Execution and Adoption (See Figure 15).

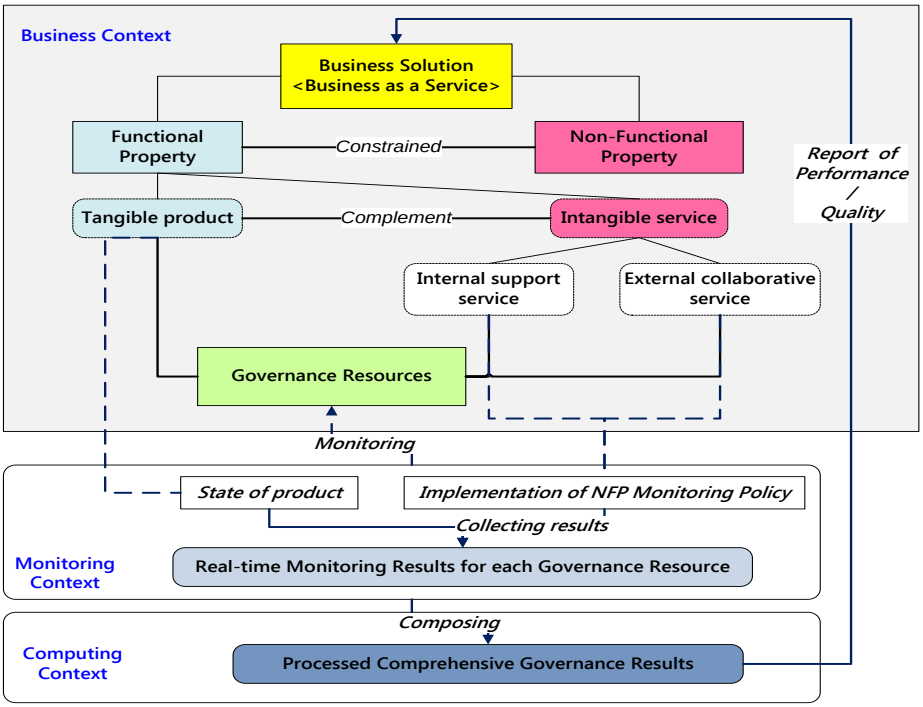


Figure 14 *Global View of Business as a Service Governance in Product Service Industrial Context*

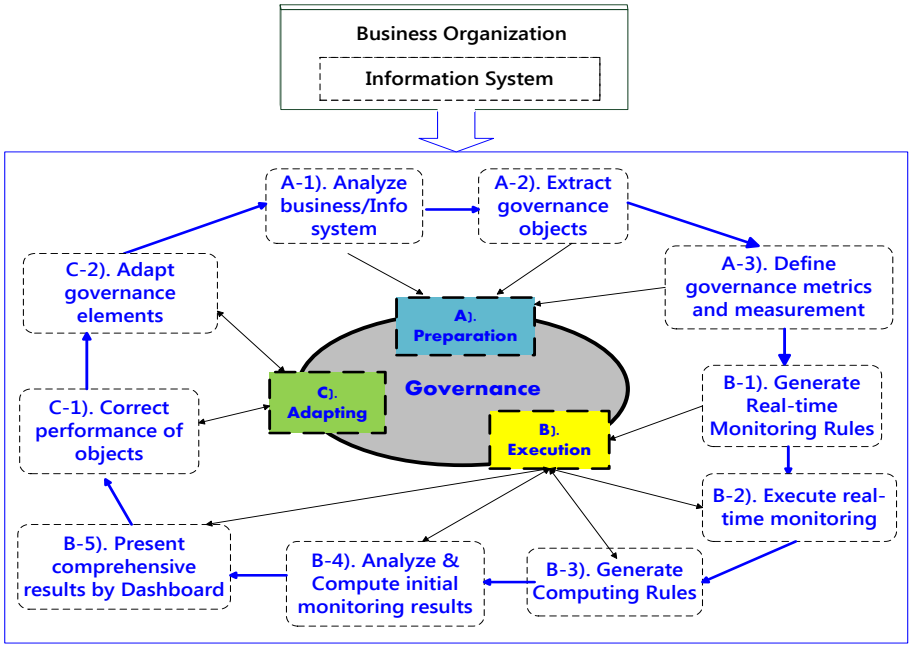


Figure 15 *Global View of Governance Loop*

The preparation phase (named phase A) is used to prepare governance required elements, paying attention on business organization. Business Decision Makers raise QoS-based business process management requirement. Service Oriented Management Governance Architecture has to accept and formalize these requirements and organize available business resources to build customized business scenario. Then all governance required documents, such as resource organization, governance agreements, Non Functional Property classification are prepared. In order to fit our multi-layer governance organization, we extend the concept of Service Level Agreement (SLA) to Multi-Level Agreements (MLAs). MLAs include Business Process Level Agreement (BPLA) and Business Service Level Agreement (BSLA). These MLAs specify the definition of business value and business needs. They also define participants' obligations, interests and punishment. This multi-level agreement strategy aims to reduce the complexities of negotiation and the number of re-configuration for Business Decision Maker. All these activities are split into 3 steps:

A-1). Analyzing business and information system: it aims at understanding the business context and the interdependencies of involved resources,

A-2). Extracting governance objects: it aims at extracting the critical success factors, and then define them as governance objects,

A-3). Defining governance metrics and measurement: it aims at reaching multi-level Governance Agreements which cover all required governance metrics and measurements.

In order to achieve an automatic management of these agreements, we design a 6-phase lifecycle for them (see Figure 16):

(a). Agreement template development: it is used to develop customized agreement template according to specific business requirements;

(b). Negotiation between business solution provider and consumer: it aims to reach agreement about governance metrics, measurements and other governance content;

(c). Achieving the governance agreements integrating elements from previous phase (a) and (b);

(d). Execution governance actions to implement achieved agreements;

(e). Assessment of governance results picked from the governance actions;

(f). Termination: ends of the agreement's lifecycle. The governance instance is completed, and them we can "terminate" the governance agreement instance.

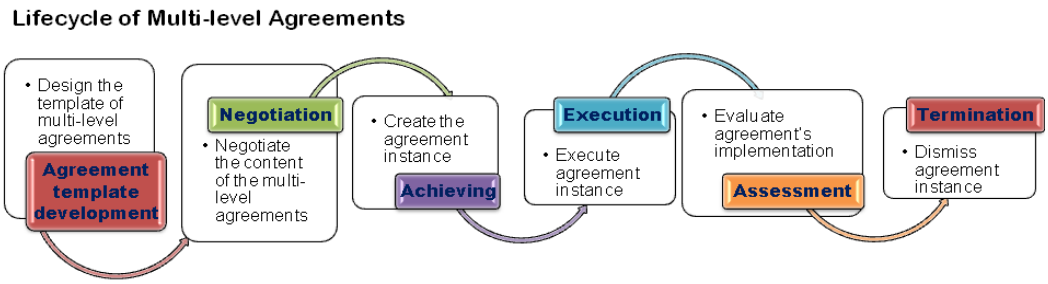


Figure 16 *Lifecycle of Multi-level Agreements*

After preparation phase, the Governance Execution phase (named phase B) aims at generating all formative Governance Policy Rules (include Real-time Monitoring Policy Rules and Computing Rules) according to the accepted governance requirement. To implement these Monitoring Policy Rules, convenient Key Performance Indicators are deployed using Governance Patterns (GPat). Computing Rules are implemented thanks to convenient Aggregators that are invoked according to required Critical Success Factor and resources' composition types. Governance Execution phase which generates governance rules and orchestrates governance components to implement governance requirements includes five steps:

B-1) Generating Real-time Monitoring Policy Rules: it consists in eliciting Monitoring Requirements, generating automatically customized Monitoring Policy Rules in order to implement these Monitoring Requirements;

B-2) Executing Real-time monitoring process: it aims at deploying governance elements (Key Performance Indicators: KPIs) to implement generated Monitoring Policy Rule,

B-3) Generating Computing Rules: it consists in eliciting Computing Requirement before generating automatically Computing Rules to implement these Computing Requirements,

B-4) Executing Computing Rules to analyze and compute monitoring results: it aims at deploying governance elements (Aggregators) to implement generated Computing Rules.

B-5) Presenting processed results: presenting computing results to customizable mashup dashboard.

The last phase (called phase C) aims at adapting the organization, i.e. optimize business performance according to governance results and adjust governance parameters to fit business requirements. The unexpected performance of business resources will be corrected and the performance and quality of business process will be improved, according to the governance results. This phase includes two steps:

C-1) Correcting performance of objects: according to governance agreements and processed governance results, a trigger launches adapting elements (Action Engines: AE) to correct unexpected performance of governance objects,

C-2) Adapting governance elements: it consists in adjusting parameters of governance elements.

To achieve this three-phase governance loop, we propose a multi-layer Service Oriented Management and Governance Architecture (SO-MGA) to manage Business Process and Information System in multi-cloud cost-effectively (see Figure 17). This proposed architecture is composed of three layers containing all business, service and implementation resources, which are used to support business activities from business organization to the IT implemented components.

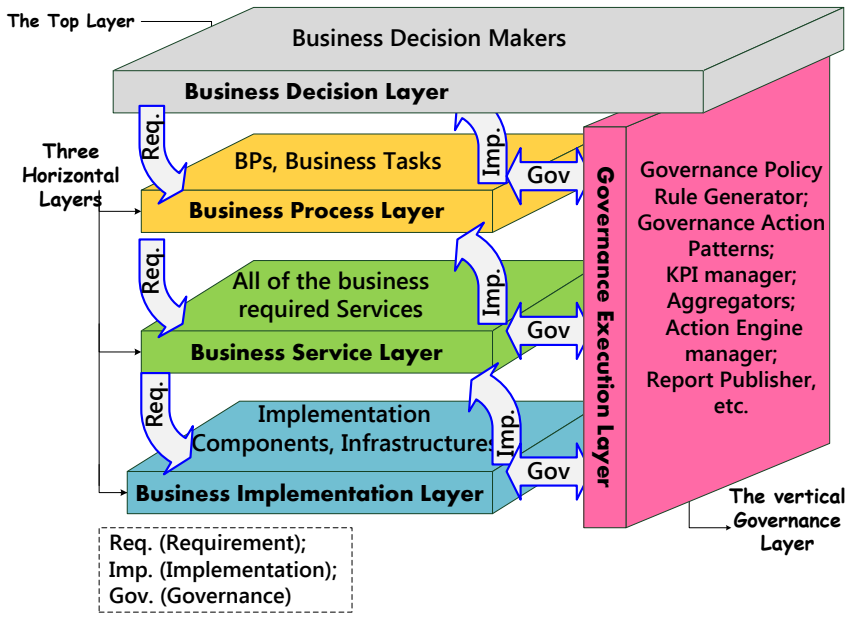


Figure 17 *Overview of Multi-Layer of Service-Oriented Management and Governance Architecture*

The Business Process Layer (BPL) is the top layer of these three "horizontal" layers. It contains all the information related to the organization of Business Processes (BPs). A Business Process can be implemented by a series of business tasks, and each business task is defined as a sub-Business Process that will be implemented using services. The Business Service Layer (BSL) contains all business tasks' required services. The Business Implementation Layer (BIL) contains all implementation components and all necessary infrastructure elements which support the implementation of services from the BSL.

Our governance loop in implemented as a "vertical" layer used to govern resources from the Business, Service to Implementation layers. The Governance Execution Layer (GEL) is used to monitor the "global quality" of the business process which deployed all involved elements in the three "horizontal" layers. As a consequence, this layer crosses all layers to deploy key performance indicators and to combine the monitoring results to a comprehensive result for the entire Business Process. The Governance Execution Layer contains all governance execution components: Governance Policy Rule Generator, Governance Action Patterns (GPats), Key Performance Indicator manager and Key Performance Indicators, Aggregators, Action Engines (AEs), and Presentation Widgets, etc.

A decision layer gathers all information from other layers to help making business decision efficiently. This Business Decision Layer (BDL) is used to keep the transparency and simplicity of governance for Business Decision Maker. It is built on the top of all horizontal layers and our governance layer. BDL contains all essential Business Decision Makers (such as financial, technical and business) to support making decision by considering all situations of others layers from both functional perspective and non-functional perspectives.

This architecture is implemented thanks to different components to support our governance loop (see Figure 18).

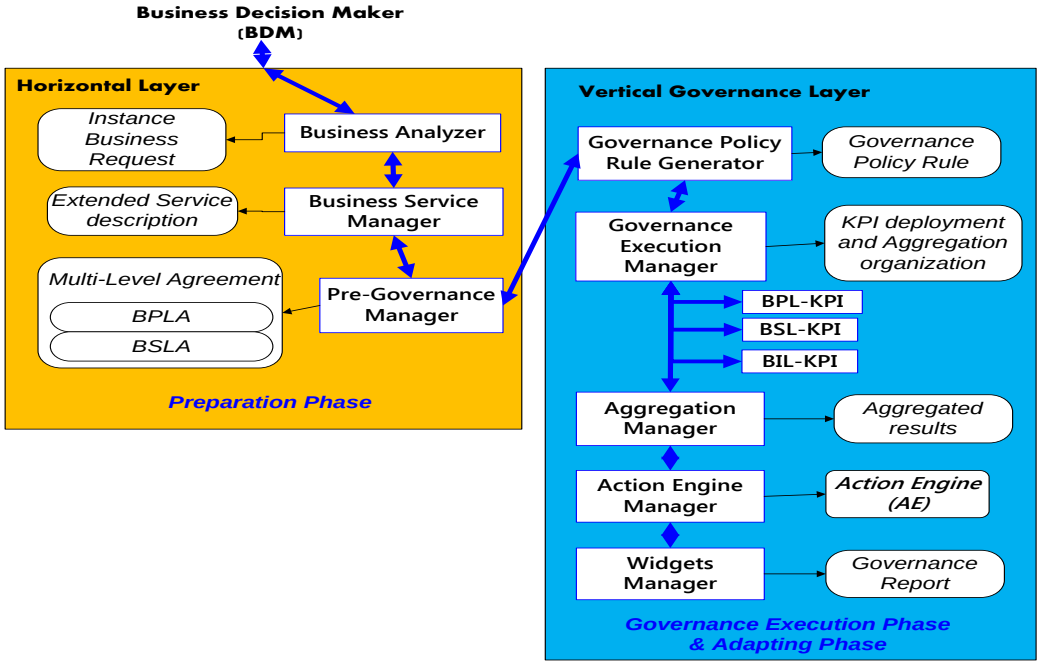


Figure 18 *Major Components of Service Oriented Management Governance Architecture*

The preparation phase is implemented thanks to 3 components shared by 3 horizontal layers:

(a) BP-Analyzer: its primary role is to accept and formalize Business Decision Maker's requirements.

(b) Business Service Manager: it has to organize available business resources and build customized business scenario.

(c) Pre-Governance Manager is used to manage governance preparation processes.

Then the Governance Execution and Adapting phase are implemented thanks to 5 components:

(a) Governance Policy Rule Generation: it has to generate governance policy rules for accepted Business Decision Maker's governance requirement, and invoke subsequent governance execution processes.

(b) Governance Action Manager: it is used to manage deployment of Key Performance Indicators and optimization processes.

(c) Aggregation Manager: it has to manage aggregation processes.

(d) Action Engine Manager: it has to manage Action Engines (AEs) to optimize performance of governed objects.

(e) Widgets Manager: it has to manage governance results' presentation widgets and dashboard.

In order to simplify the communication and information exchange between these components, we extend the traditional Enterprise Service Bus (ESB) to an Integrated Management and Governance Bus (IMGB) (see Figure 19).

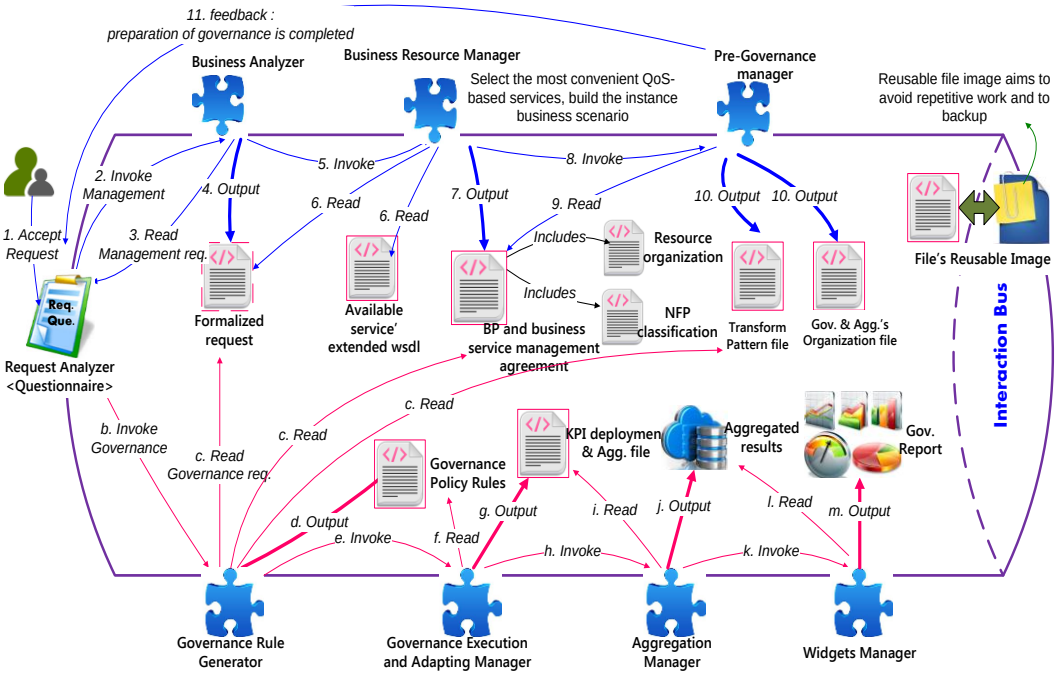


Figure 19 *Messages Exchanging in the Integrated Management Governance Bus*

This Integrated Management Governance Bus (IMGB) allows components to communicate and collaborate with each other by sharing their required predefined XML files (files' format will be detailed in chapter 4 and 5). It allows components for plug-in-and-play by adapting their input and output data format. In addition, this IMGB allows the complexity of information transport to be transparent for end-users, allowing raising requirements and checking results simply. In order to keep the processes' consistency, activities' traceability and resources' reusability, we design an instance file's image storage and reusable strategy. All created files and message exchange logs are stored as compact images which can be invoked and reused easily to avoid the replication of same instance files.

A Management and Governance Preparation Framework is in charge of management components: Business Analyzer, Business Resource Manager and Pre-Governance Manager. The interaction and information exchange between these components are shown in Figure 20. More details are presented in Chapter 4.

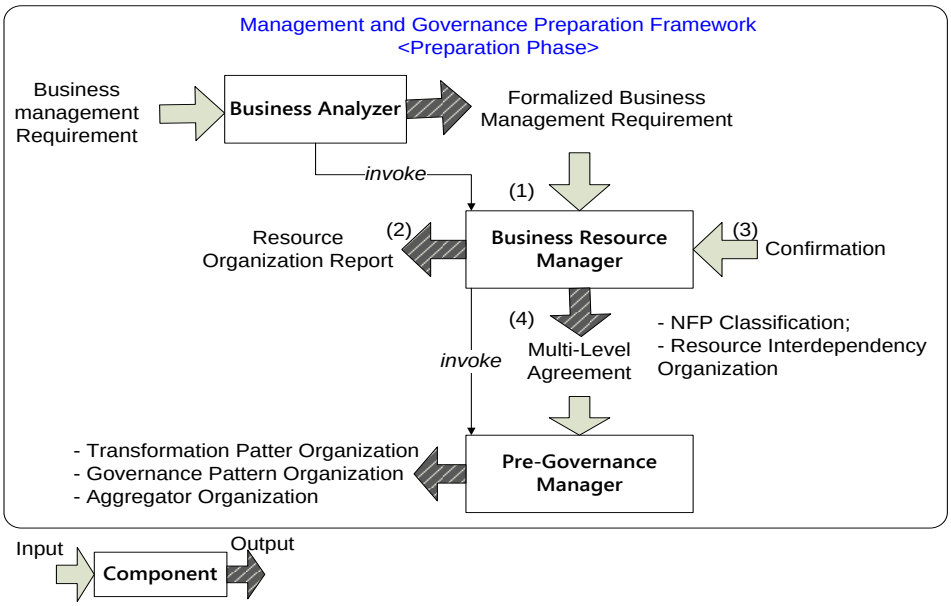


Figure 20 *Interaction of Management and Governance Preparation Components Components*

The Governance Execution and Adapting Framework is in charge of governance components: Governance Rule Generator, Governance Execution & Adapting Manager, Aggregator Manager and Widgets Manager. The interaction and information exchange between these components are shown in Figure 21. More details are presented in Chapter 5.

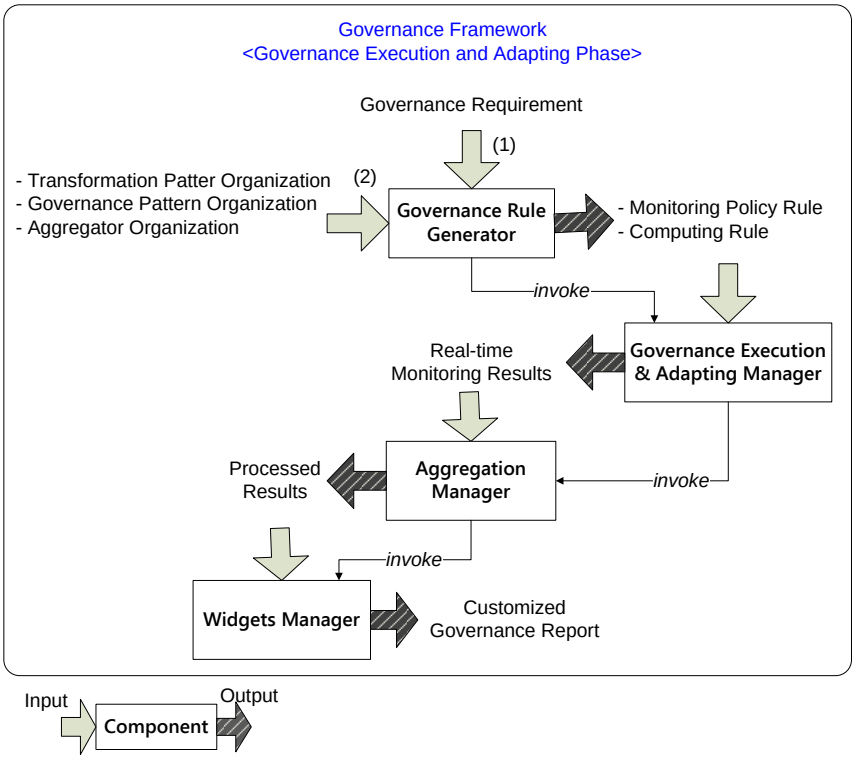


Figure 21 *Interaction of Governance Components*

3.3 Governance Property

Managing and governing BaaS involves pay attention on "Activities" and "Resources" in this service-oriented and multi-cloud business context. "Resources" of a product / service system deployed in cloudy environment include business tasks, services, operations and infrastructure components. "Activities" are defined as the consequences of Resources' actions. In addition, each resource is associated to some Functional Properties (FPs) to achieve its functional goals and to Non-functional Properties (NFPs) used to constrain the way functional properties will be reached. In a product / service industrial context, we define Governance Resource as an integration of "Tangible Product" and "Intangible Service". To maintain competency, manufactures are transforming from provide "pure products" to provide integrated solutions include products and services. Support services do not only complement and add value to products /or services but also constrain quality of products or delivered services and increase satisfaction of customers. Therefore, our governance approach pays attention on integration of product and delivered services Functional Properties and Non Functional Properties. Moreover, in nowadays

extensive collaboration environment manufacture requires cooperative support services from external partners. This trend requires the governance approach fit the collaborative environment. Consequently, a governance node has to manage its own resources Functional Properties and Non Functional Properties, it also has to support collaborative services Non Functional Properties (see Figure 22).

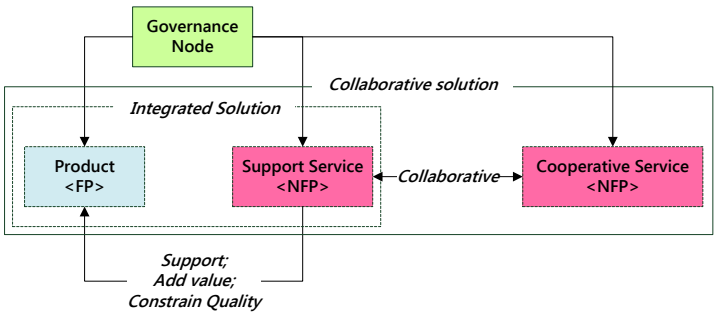


Figure 22 *Governance Orgnization for Integrated Product-Service systems*

In our study we define Business Services as all internal and external services, which are required to implement business process objectives. Business services are associated to their IT artifacts that implement them such as Business Tasks, Services, Infrastructure elements (such as hardware, equipement, database, software, etc.). These artifacts are defined as Business Resources. Business Processes aim at organizing all Business Resources and activities to achieve a common organizational objective.

Our architecture has been designed to map or derive business processes into a business service chain which is complemented by composing infrastructures elements. More precisely in a top-down workflow implementation perspective, a Business process is organized as a composition of different Business Tasks. A Business Task can be either seen as a sub-Business Process (sub-BP) or as an elementary task. Each elementary Business Task (further named Task) requires one or more service's collaboration to achieve its functional requirements. Each service requires one or more operations to achieve its functional goals. Then each operation requires a composition of infrastructure components to be executed. Non Functional Properties are also associated to each of these "functional" elements to constrain the way they may be implemented or executed (see Figure 23).

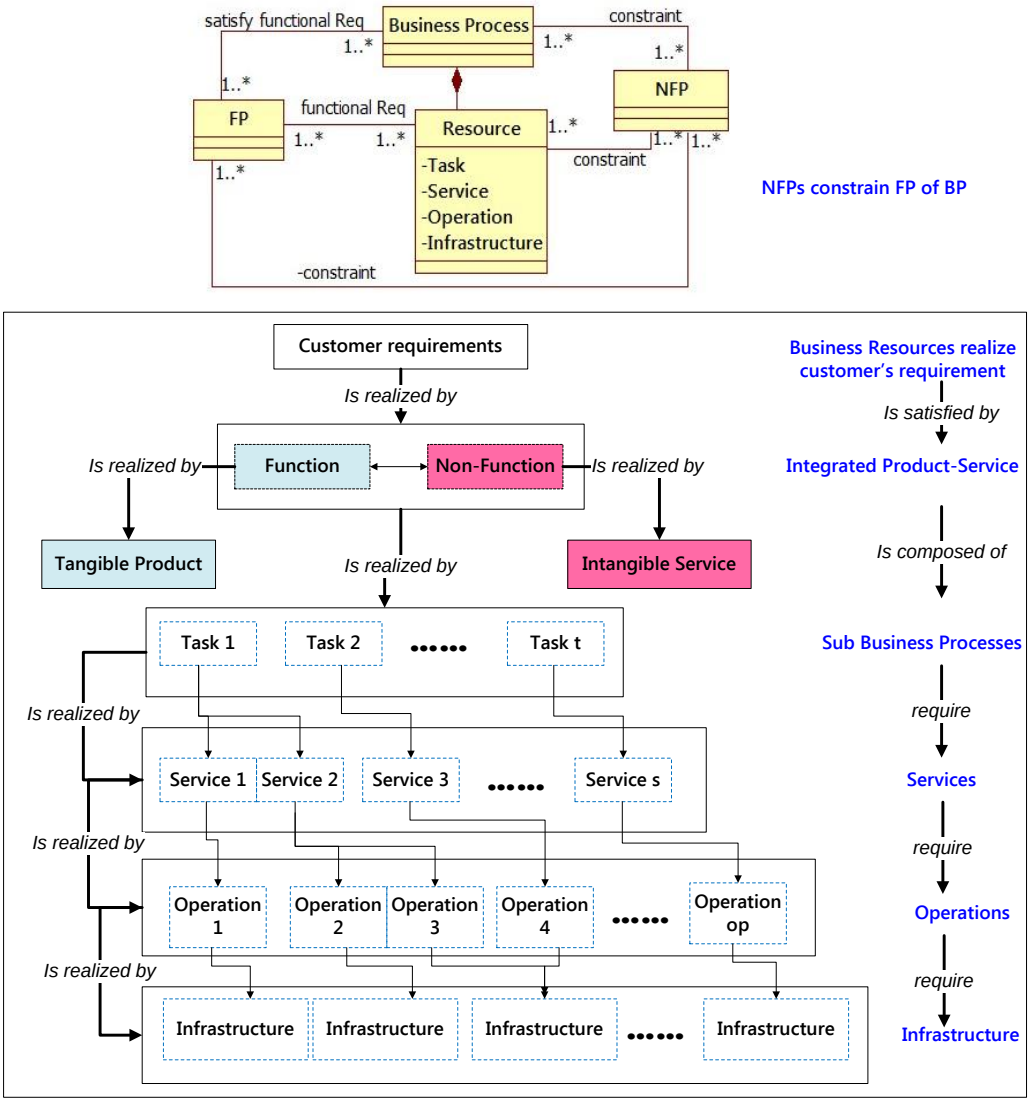


Figure 23 *NFPs Constraint FPs and Resource Mapping Plan*

As we focus on the IS value chain governance, we use the IS artifacts associated to the different industrial resouces. As a consequence, our global resource model is designed to focus on Business Process and its implementation, taking into account five types of resources picked from the IT value flow: Business Process (BP), Business Task (Task), Service, Operation and Infrastructure. Formally speaking the resource set is defined as

Resource = BP ∪ Task ∪ Service ∪ Operation ∪ Infrastructure, R=Resource. (Equation 1)

Each resource R is identified and distinguished by its functional and non-functional properties. Then the quality of Operations impacts the Quality of Service (QoS). QoS can impact the quality of a Task, whereas the associated Tasks' quality impacts the Business Process quality. Integrating the links between resources leads us to define a Resource in our governance architecture as a tuple:

$Resource\_k$= (ID, Type, Goal, Context, Endpoint, FP, NFP, RequiredResources)

(Equation 2)

Where

- ID defines resource's identity;
- Type defines resource's type whether it is a white-box fully controlled resource or it is a black-box resource that just need to pay attention on input and output;
- Goal defines the objectives of this resource;
- Context defines the resource's implementation context;
- Endpoint defines the resource endpoint, i.e., an entry for accessing this resource;
- FP defines the functional properties of this resource which are provided by the physical product or the delivered service;
- NFP defines the non-functional properties of this resource which constrains the achievement of the Functional Property of this resource. In our governance process we pay a particular attention on Product Support Service and Collaborative External Service related Non Functional Property.
- RequiredResource defines required resources to implement this resource. It expresses the dependency relationship of this resource with other resources.

Depending on the layer a resource belongs to, some extra elements can be added to precise this generic model (see chapter 4 and 5).

According to this definition, we can gather all the resources as:

$Rs = \{R\_k\}$ where $0<k<=Nk$; (Equation 3)

Where "k" is the resource number and Nk is the total number of resources.

The following figure shows the excerpt of the schema of resource's definition (see Figure 24):


```
11  <xs:element name="Resources">
12    <xs:complexType>
13      <xs:sequence>
14        <xs:element ref="Resource" maxOccurs="unbounded"/>
15      </xs:sequence>
16      <xs:attribute name="layer" type="xs:string" use="required"/>
17    </xs:complexType>
18  </xs:element>
19  <xs:element name="Resource">
20    <xs:complexType>
21      <xs:choice maxOccurs="unbounded">
22        <xs:element name="ID" type="xs:string"/>
23        <xs:element name="Type" type="xs:string"/>
24        <xs:element name="Goal" type="xs:string"/>
25        <xs:element name="Context" type="xs:string"/>
26        <xs:element name="Endpoint" type="xs:string"/>
27        <xs:element name="FP" type="xs:string"/>
28        <xs:element ref="NFP"/>
29        <xs:element ref="RequiredResources"/>
30      </xs:choice>
31    </xs:complexType>
```

Figure 24 *The Excerpt of Resource's Definition*

To overcome challenge of non functional requirements elicitation, specification, interdependencies and clustering, our governance solution takes into account the business functional organization and we pay attention on Non Functional Properties which constrain the quality of Functional Properties. We organize our monitoring and computing processes according to business resources' organization. Therefore it allows governance elements to be composed and orchestrated while running the business processes and allows governance execution and adaption actions fit business goals.

In order to specify governance for integrated product-service systems, we divided governance property into two directions: (1) Functional Property focuses on quality of product, (2) Non Functional Property focuses on quality of service and performance of support systems. Each group can be divided into more detail sub-groups according to governing needs. Finally each Non Functional Property is specified into Critical Success Factor sets associated to metrics used to constrain and / or evaluate its accomplishment. All required Non Functional Properties are defined in Governance Agreements to constrain associated resources.

In our research we pay attention on both industrial performance indicators and IT quality indicators to propose an integrated performance/quality indicator management system. We pick up the most concerned Non Functional Properties from performance, maintainability, cost and security aspects. These

governed Non Functional Properties are classified into four groups based on their features according to business needs. Performance group focuses on availability rate, delay rate, response and execution time. More precisely, delay rate consists in response delay rate and execution delay rate. Maintainability group focuses on reputation, reliability, usability and accuracy. Cost group focuses on price. Security group focused on non-reputation, confidentiality and integrity.

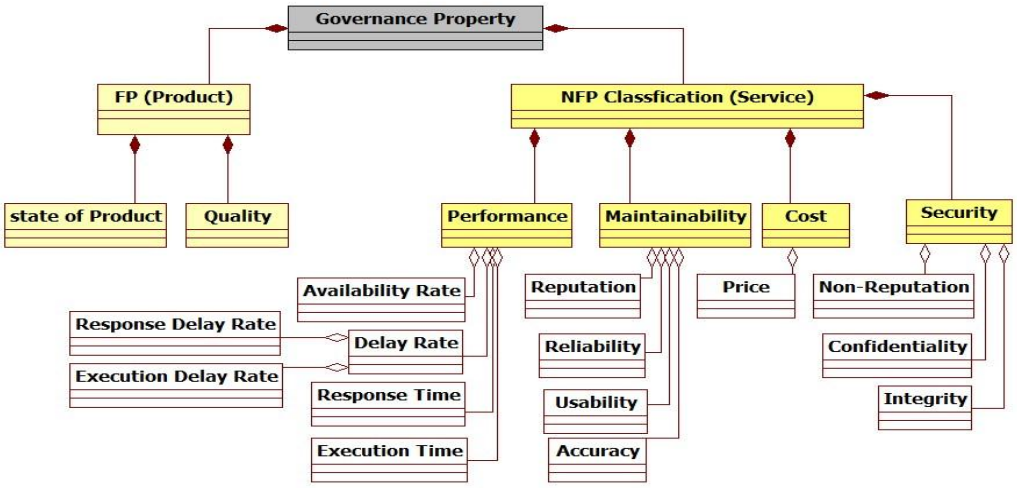


Figure 25 *An example of Governance Property*

In order to achieve an automatic management and governance systems which can minimize human intervention, we design five phases Resource's lifecycle to organize and optimize the usage of resources (see Figure 26). This lifecycle includes:

1) "Preparation", in this phase resources are prepared to be selected and composed;

2) "Activation", in this phase required resources are composed and orchestrated to complete required business scenario;

3) "Execution", in this phase resources are run functionally to implement the business scenario which has been built in the previous phase.

4) "Evaluation", in this phase the quality and performance of resources' non-functional properties are evaluated. The evaluated results are used to select the resource's next lifecycle phase.

5.1) If the resource can be reused or it meets the requirement of business management "Modulation" phase is launched. After a modification process, this resource can be reused and its lifecycle restarts from first phase "Preparation".

5.2) If the resource's assessment result shows the resource cannot meet the business requirement or if it will not be used again, "Decommission" phase is launched.

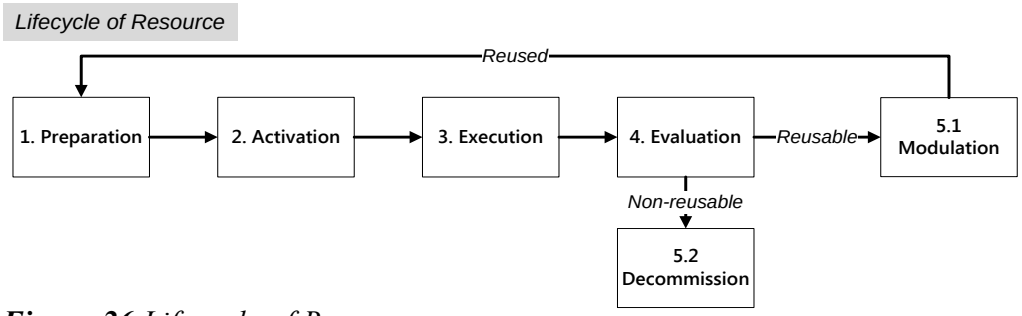


Figure 26 *Lifecycle of Resource*

3.4 Governance Elements' Organization

Our Service Oriented Management and Governance Architecture (SO-MGA) takes advantage of the MDE strategy and of the pattern-based engineering strategy. It takes into account Non Functional Property to generate series of customized Monitoring Policy Rules and Computing Rules. It also integrates a "model at runtime" vision to satisfy governance requirements flexibility. In this section we introduce the governance elements (Governance Rules, Key Performance Indicators (KPI), Patterns, etc.). Details of these governance elements organizations will be introduced in chapter 4 and 5.

The Model-Driven Engineer (MDE) strategy is used to generate two types of Rule automatically.

- Monitoring Policy Rules (PolRs) are generated to implement Monitoring requirements;

- Computing Rules (CompRules) are generated to compute initial monitoring results in order to implement comprehensive governance requirement.

To generate these rules, we've designed two types of patterns are designed to transform requirements into governance rules and then to deploy governance elements used to implement the governance process. This governance process includes a monitoring process and a computing process. These two types of patterns are:

- Transformation Pattern (Pat) aims at transforming monitoring requirement into monitoring rules.

- Governance Pattern (GPat): GPat aims at transforming Monitoring Policy Rule's information to convenient Key Performance Indicators (KPIs) and

deploying convenient Key Performance Indicators to implement the generated monitoring rules.

Our generated governance rules (PolRs and CompRules) uses two types of essential governance elements: Key Performance Indicator and Aggregator.

- The Key Performance Indicator (KPI) aims at implementing the generated real-time Monitoring Policy Rule by measuring and recording monitoring object's performance and quality.

- The Aggregator aims at producing comprehensive governance results accordingly by applying the appropriate computation algorithm to the collected monitoring results in a real time way.

Then, to achieve autonomic management, we design a 6-phase lifecycle for these governance elements (see Figure 27):

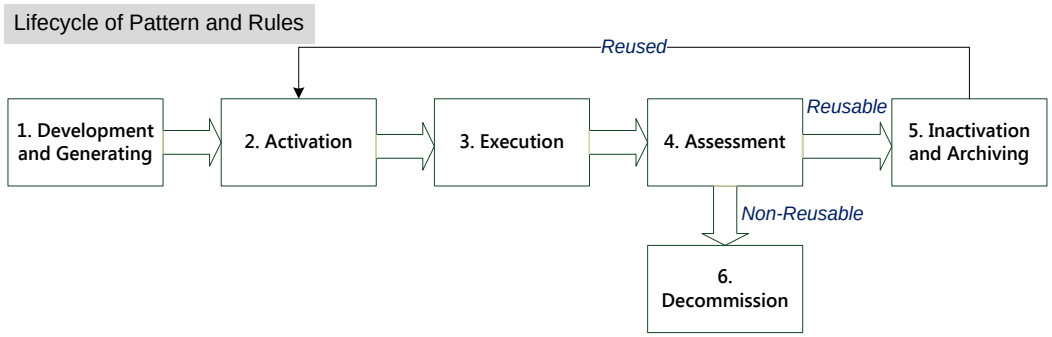


Figure 27 *Lifecycle of Patterns and Governance Rules*

1) "Development and Generating", in this phase patterns are developed and governance rules are generated;

2) "Activation", in this phase, patterns and governance rules are invoked;

3) "Execution", in this third phase, patterns and rules are executed and to implement governance requirements;

4) "Assessment", in this phase, the implementation performance of patterns and governance rules are evaluated to decide their next lifecycle phase.

5) "Inactivation and Archiving", if the result evaluation proves that the pattern or the governance rule can be reused. In such case the elements inactivated and archived. Archived patterns and rules can be reused when they are invoked by the activation step.

6) "Decommission", when the pattern and/or rule cannot be reused, a termination process is launched to end element's lifecycle.

3.5 Management and Governance Working Principle

As we have mentioned before, our Service Oriented Management Governance Architecture has two main objectives: 1) providing and managing Non Functional Property and QoS-based Business as a Service (BaaS), 2) governing and evaluating the performance and quality of BaaS. In this section, we globally introduce our governance architecture's working principles (see Figure 28).

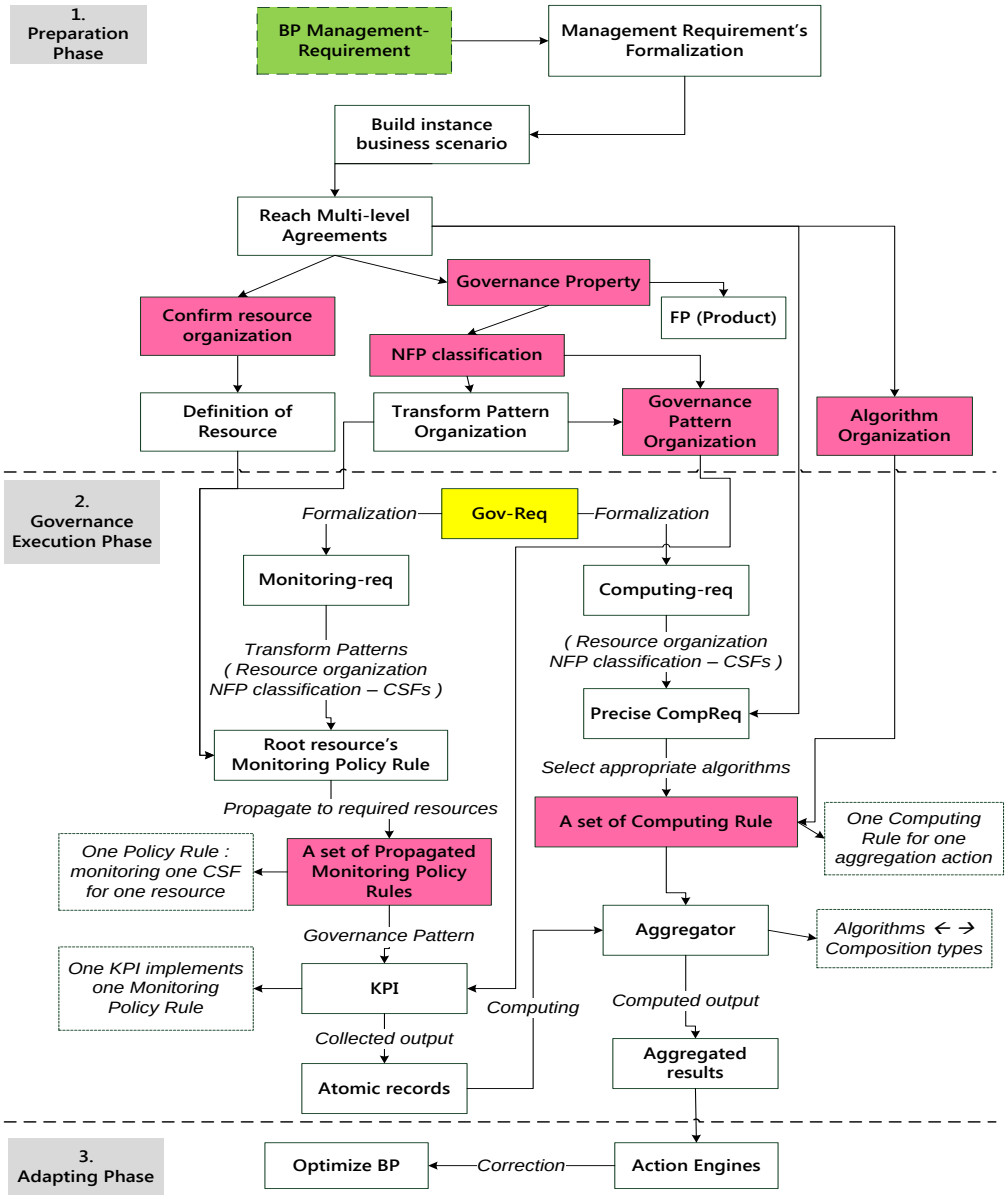


Figure 28 *High level Working Process of Multi-layer BaaS Management and Governance Architecture*

3.5.1 Business Resource Management and Governance Preparation

The framework we design to support the Business as a Service management and governance preparation phase is designed as a middleware and coordinates the Business Decision Maker, service providers and infrastructure elements. It aims at managing multi-level agreements to reduce wastes as well as the number of agreements' violation. It organizes business resources efficiently and effectively to increase Business Decision Maker's profit.

The BaaS management and governance preparation process includes four steps:

1) Accepting Request: the Business Decision Maker can raise a Non-functional Requirements (NFRs)-based Business Process management request to our SO-MGA. After receiving NFR-based request from Business Decision Maker, our framework creates an instance of requirement to implement this request and invokes the Request Analyzer.

2) Analyzing Request: after receiving Business Decision Maker's Non Functional Property-based management requirement (Mgm-Req), the Request Analyzer formalizes this Mgm-Req to specify Business Decision Maker required Non Functional Properties. It reorganizes required Business Process as a series of specified sub-Business Processes associated to the required Non Functional Properties. Then the Request Analyzer creates the business request instance used to invoke Business Service Manager to build the required business scenario.

3) Building Business Scenario: the Business Service Manager uses the generated request instance to select the available services and other business resources to build required business process. Meanwhile, it negotiates with the involved service providers and establishes a multi-level agreement (which includes Business Process Level Agreement and Business Service Level Agreement) to constrain the quality of business process. Building the required business scenario and the multi-level agreements include Resource Organization and Non Functional Property classification. The pre-Governance Manager is invoked to complete governance preparation process.

4) Preparing Governance: based on the generated multi-level agreements, the documentations of governance elements (the Governance Transformation Pattern, the Governance Pattern and the Aggregator-Algorithm) are generated in this step.

To sum up, this management and governance preparation process will generate the following files that will be used by further governance loop:

1. Multi-level Agreements: it includes Business Process Level Agreement; Business Service Level Agreement;
2. Registered Non Functional Property classification;
3. Resource dependency organization;

4. Transform Patten organization;
5. Governance Pattern organization;
6. Key Performance Indicator organization;
7. Aggregator organization.

The concrete processes of this management and governance preparation model will be introduced in detail in chapter 4.

3.5.2 Governance Execution and Adapting

To support the Governance Execution phase and Adapting phase of our governance loop, we propose a Governance Execution and Adapting Framework. It aims at implementing dynamic runtime governance processes which can narrow the gap between business and technology. It takes customized governance requests to constrain the quality of specified Non Functional Properties, and provides detailed and comprehensive governance results. It is also used to improve the performance of business process using a pattern-based transformation process. Received governance requirements are turned into Platform Independent Policies and Platform Dependent policies used to orchestrate the Non Functional Property management components at runtime. Moreover, we take advantage of the functional specification of Resource Action Composition to compose the Non Functional Property orchestration, governance policies and monitoring results accordingly.

As in the preparation phase, the policy generation process takes advantage of both Model Driven Engineering and of Pattern-based Engineering approach. The transformation strategy relies on a global model that connects business workflow analysis and governance requirements to governance patterns and policy rules. Governance requirements are divided into Monitoring Requirement and Computing requirement. Each monitoring requirement is analyzed and transformed into monitoring policy rules thanks to 'transformation pattern'. Each computing requirement is analyzed and transformed into computing rules thanks to 'computing pattern'. Generated monitoring policy rules select, orchestrate and invoke 'Governance pattern' to constrain accomplishment of functional rules by assigning and computing 'Key Performance Indicators'. Generated computing rules select, orchestrate and invoke convenient 'Aggregator' to compute Key Performance Indicator's monitoring result by algorithms. According to the governance results, an 'Action Engine (AE)' can be invoked to correct and improve the resources performance by doing actions on the related resources. All the results can be published as mashup reports depending on users' needs.

This governance execution and adapting process includes 7 main steps:

1) Receiving and formalize monitoring requirement: in this step, Business Decision Makers raise real-time monitoring requirement (MonReq) to SO-MGA. SO-MGA formalizes Business Decision Maker's monitoring requirement and then invokes Governance Policy Rule Generator to generate monitoring policy rules (PolR).

2) Generating Monitoring Policy Rule: in this step, the monitoring policy rule generator parses the received MonReq to extract the required Root Resources and Root Non Functional Properties. After that, the refinement process is launched to call the convenient Transformation Patterns. This refinement process is launched recursively until it gets all the required Critical Success Factor (CSF) patterns from the Root Non Functional Properties. Therefore Root Resource's monitoring rules are generated. Using the business organization knowledge, related resources are extracted thanks resource dependency organization. Then the Root Resource's basic policy rule is propagated to all related resources.

3) Invoking Governance Elements: Key Performance Indicators (KPIs) are invoked by the Governance Patterns (GPat) selected thanks to the generated Monitoring Policy Rules (PolRs).

4) Formalizing and Parsing Computing Requirement: the similar way as in step one, in this step, the Computing Requirements are formalized into predefined templates. Then these requirements are parsed to extract required Root Resource and Root Non Functional Property.

5) Generating Precise Computing Requirement: specifying computing requirement thanks to resource refinement process and Non Functional Property refinement process. A resource refinement process is launched to extract related resources and a Non Functional Property refinement process is launched to extract related Critical Success Factors. Therefore Precise Computing Requirements are generated.

6) Generating Computing Rules are used to select convenient Aggregators comparing Critical Success Factor type and Resource composition type. Lastly Computing Rules (CompRules) are generated.

7) Implementing Computing Rules: the Aggregators are used to compute in real-time the Critical Success Factors' monitoring results. Action Engines (AEs) can be invoked to act on target resources to correct or improve the performance of that resource accordingly. After this computing step, processed results are presented by different widgets. These widgets are organized in customized way in a presentation dashboard.

Details of these steps will be introduced in chapter 5.

3.5.3 Introduction of Use Case

For demonstrating continuously our governance solution, we take a Logistics Company as a Use Case. This company's functional objective is to distribute goods from Factories to End Clients. It requires different delivery means (such as by road, by air, by train, etc.), choses different distribution nodes (such as depot, distribution centers, transmit points, etc.) to organize business process instance to meet difference business needs. This Use Case is used gradually through our contribution chapters (see Figure 29).

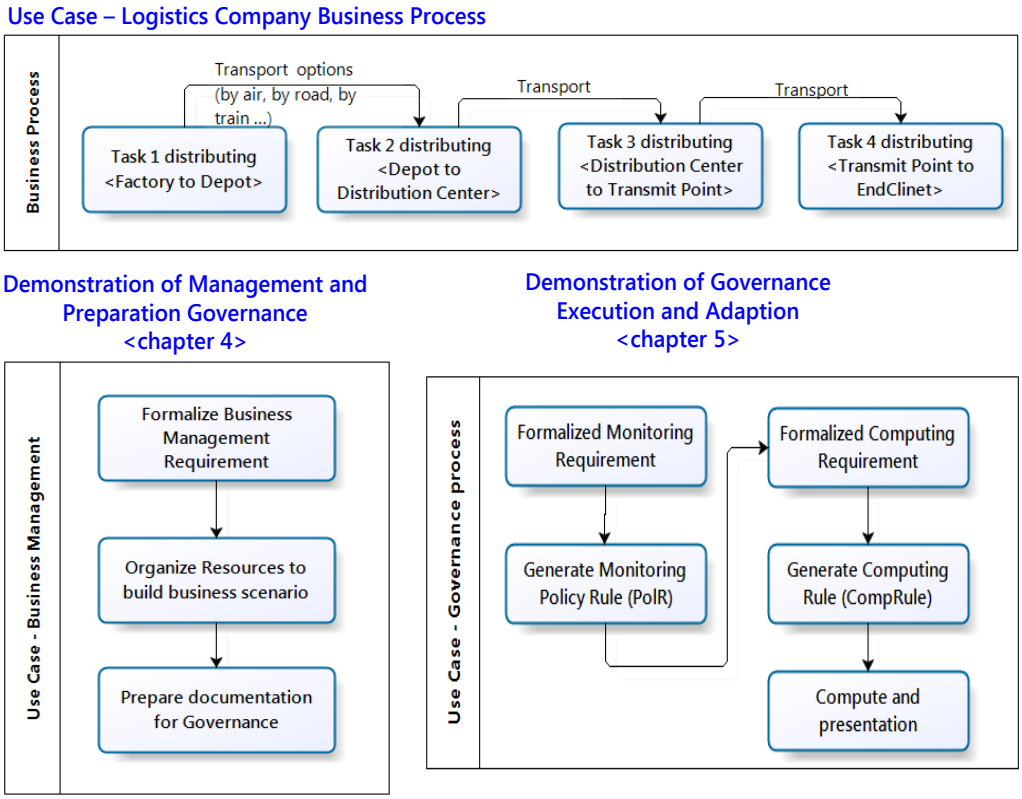


Figure 29 *Use Case Organization*

3.6 Conclusion

This chapter gave a global view of our multi-layer Service Oriented Management and Governance Architecture (SO-MGA) and introduced the working principles of our SO-MGA and its major components. In a nutshell, to get benefits and to overcome the limits of service-oriented multi-cloud context, it requires a common cloud service reference architecture enabling cloud service

management and governance to integrate QoS and Non Functional Property in a transparent way at runtime. By now, the different works do not cover these requirements. The gap between business layer and technical layer is still a challenge that needs to be solved. To this end, we propose a governance loop which includes following three main phases: Preparation, Execution and Adoption. To implement this governance loop, we propose a multi-layer Service Oriented Management and Governance Architecture (SO-MGA) which takes advantage of the service-oriented multi-cloud deployment to support collaborative business requirement integrating a unified approach to deploy Non Functional Property-based Business as a Service management and integrated QoS and Non Functional Property performances governance from business perspective to infrastructure in a dynamic way.

The governance process also can be taken as a "Governance as a Service". All governance components are composed and orchestrated while running the business process by the Integrated Management and Governance Bus (IMGB) strategy. This IMGB-oriented strategy takes advantage of service-oriented loosely coupled with minimum runtime footprint over the execution environment, to fit the elasticity and transparency requirement.

To implement the governance loop and the Service Oriented Management and Governance Architecture, we propose two frameworks: 1) Business as a Service Management and Governance Preparation Framework; and 2) Business as a Service Governance Execution and Adapting Framework. In the following chapters we will give more details about our Service Oriented Management and Governance Architecture.

4 Business as a Service Management and Governance Preparation Framework

4.1 Introduction

As introduced in Chapter 3, this chapter focuses on the implementation of the first phase (Preparation Phase) of our Governance Loop and introduces our Business as a Service (BaaS) Management and Governance Framework.

As stated in section 1.2 (Research Issues and Contribution), the governance approach we develop to fit the product service agility and high quality is based on twe questions:

- Q1: what should be governed?

- Q2: how the governance objects are organized and what is the interdependence of these objects? :

The first question means that we have to understand what are the most important factors impact business performance. In product-service industry, enterprises not only produce pure product for customer, they also provide support services to increase customers' satisfaction and improve their competitiveness. In other words, enterprises produce integrated business solutions instead of pure product. Therefore the quality of integrated business solution (including product and support services) is the critical success factor for business performance. We set the quality of integrated business solution as our governance object and due to the business organization every related products and service are considered as governance objects. In our approach Non-functional Properties (NFPs) constrain the quality of governance objects. As a consequence we pay attention on Non Functional Properties' governance. Furthermore, as we have discussed in State of the Art (section 2.3) elicitation of Non-Functional Requirements requires consideration on Functional Properties (FPs) and on business organization due to our industry context.

To answer the second question, i.e. identify how governed objects are organized and their interdependencies, we have to take into account the business organization. Understanding the interdependence among these governance objects and interdependence between Non Functional Properties and Functional Properties, requires a Non Functional Property clustering method. To this end we propose a Non Functional Property classification method to specify Non Functional Properties, and take into account the interdependence between Non Functional Properties and Functional Properties by building resource

dependency organization. Furthermore, in this service oriented multi-cloud environment, Business Processes are complex and collaborative. A Business Process often requires a variety of services. These services might be selected from difference Service Providers (SPs) in different domains. As a consequence, negotiation and management of Service Level Agreements (SLAs) become an onerous work for Business Decision Makers. Any violation of these SLAs could lower the quality of entire Business Process. To free Business Decision Makers from those arduous works and take advantage of multi-cloud, we propose a customized transparent SLAs violation management strategy to reduce the harm from SLA violation.

To solve these research questions we propose a Business as a Service (BaaS) Multi-layer Management and Governance Preparation Framework. This framework aims at efficiently organizing business resources, managing business resource and relevant Agreements, as well as preparing governance. It allows Business Decision Maker to make decision efficiently, seize business opportunities and maximize business' profits.

This BaaS Management and Governance Preparation Framework is composed of two models:

1) The Business Resource Organization Model (BROM) aims at providing a QoS-based business solution to satisfy Business Decision Maker's specific request. To achieve this goal, we propose a two steps process:

- "Translation", it aims at deconstructing and formalizing Business Decision Maker's requirement from Business Side Language into Technical Side Language.
- "Organization", it aims at selecting and organizing required business resources.

Then our resource organization model compares the selected convenient Services to generate a comparison report for Business Decision Makers to choose the most appropriate Services. Once Business Decision Makers confirmed the choice, connections between Business Decision Makers and required Service Providers are created.

2) The Negotiation and Governance Preparation Model (NGPM) aims at narrowing the gap between business requirement and technical specifications, and simplifying the negotiations between Business Decision Makers and Service Providers (SPs). Achieving cross-layer governance, we extend the traditional Service Level Agreement (SLA) to Multi-level Agreements (MLAs). These MLAs include two agreements: Business Process Level Agreement (BPLA) and Business Service Level Agreement (BSLA). BPLA is the agreement signed between Business Decision Makers and our Service Oriented Management and Governance Architecture (SO-MGA) to define all promised Functional Properties, Non Functional Properties, obligations and penalties. The BPLA is

designed according to a business perspective. It allows Business Decision Makers to understand and manage the collaboration easily. BSLA is the technical perspective agreement signed between SO-MGA and Service Providers. BSLA can be seen as a technical version of BPLA. In order to guarantee the promised QoS, reducing the cost and number of reconfiguration caused by Agreement violations, we defined a Quality Range (QR) model for BPLA and BSLA. The Quality Range defines thresholds split the quality to a resource into 3 different situations: Satisfied, Tolerable and Alert situation. Quality Range aims at managing precisely quality of resource. It gives chances to correct the performance of BaaS and reduces the probability of paying penalties. All in all, these MLAs allow Business Decision Makers to manage business resources efficiently and simply. In the following subsections we first introduce the specification of BPLA, Quality Range and BSLA. Then we introduce our Governance Preparation Process defining and organizing the essential governance elements. The logistic consistent Use Case introduced chapter 3 is used for demonstrating the working processes of this proposed BaaS Management and Governance Preparation Framework.

This chapter is organized as follow: in the section 4.2 presents the Business Resource Organization Model, including the Deconstruction of Management Requirement (section 4.2.1), the Formalization of Deconstructed Requirement (section 4.2.2), the Selection Business Process (section 4.2.3) and we present the Use Case structure and implementation (section 4.2.4 and section 4.2.5). Section 4.3 will detail the Negotiation and Governance Preparation Model, including Business Process Level Agreement (section 4.3.1), Business Service Level Agreement (section 4.3.2), Governance Preparation-Classification Non Functional Property and Resource Dependency (section 4.3.3), Governance Preparation-Definition of Transformation Pattern and Governance Pattern (section 4.3.4), Governance Preparation-Defintion of Key Performance Indicator and Aggregator (section 4.3.5) and Use Case-Definition and Dependencies (section 4.3.6), Use Case-Governance Preparation (section 4.3.7 and section 4.3.8).

4.2 Business Resource Organization Model

The Business Resource Organization Model (BROM) is designed to build customized Business as a Service (BaaS), which can optimize organization of available business resources to fit Business Decision Makers' customized business process requirement.

In product-service systems, business resources are integrated business solutions which include functional properties (product) and non-functional

properties (support services). A Management Requirement (Mgm-REQ) is deconstructed into several detailed Task Requirement (Task-req). Then, this Task-req is also deconstructed into one or more Service Requirements (BS-reqs) that are used to select convenient business resources. After selecting convenient business resources, a Business Process Organization Report is generated. Business Decision Makers (BDMs) will use it to support their decision. After Business Decision Makers confirm the business process organization acceptance, the customized business scenario is built. Figure 30 presents this process.

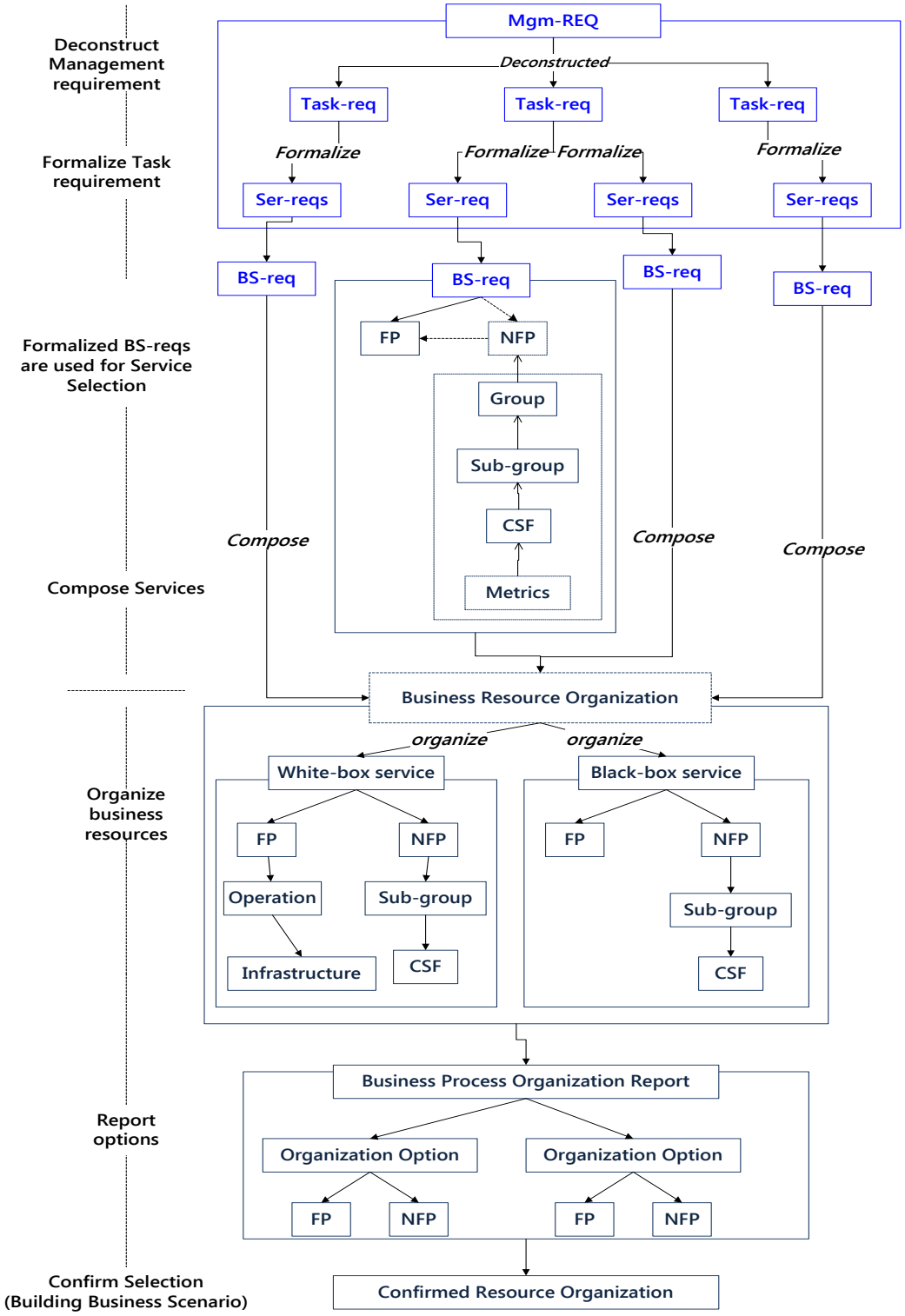


Figure 30 *Business Resource Organisation Model*

As stated section 3.2, Business Analyzer and Business Resource Manager are driven by this Resource Organization Model. Figure 31 shows the interaction and information exchange between these components.

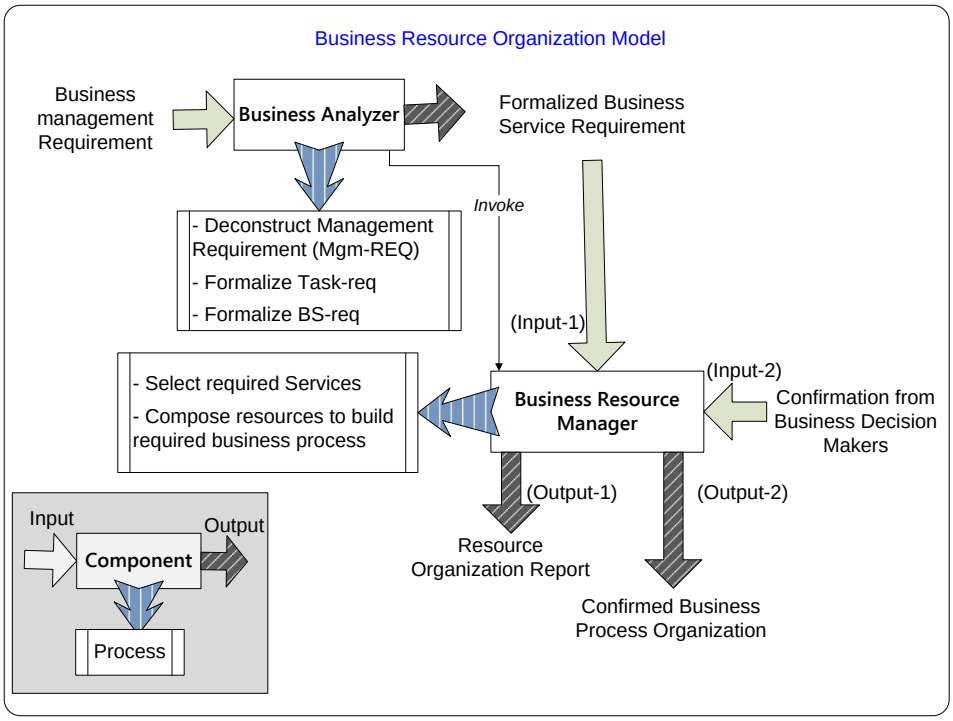


Figure 31 *Interaction of Organization Components*

4.2.1 Deconstruction of Management Requirement

The starting point is the reception of a Management Requirement (Mgm-REQ). This requirement is sent by the Business Decision Maker. It is deconstructed into a series of Business Task Requirements (Task-reqs) to be implemented precisely. A Task-req is defined as a tuple:

$$\text{Task-req}_{tr} = (\text{Task-reqN}, \text{Task-reqG}, \text{Task-reqS}, \text{Task-reqNFP}, \text{Task-reqCxt}), \quad \text{(Equation 4)}$$

Where

- Task-reqN defines this Task name;
- Task-reqG defines this Task's objective;
- Task-reqS defines which service requirement this Task-req requires.
- Task-reqNFP defines required Non Functional Properties to constrain this task execution.
- Task-reqCxt defines this Task's implementation context.
- 'tr' is the Task-req number.

We can gather all Task-reqs associated to a same business process requirement (BP-req) thanks to a selection function σ. This selection function matches the Task-reqG with BP-req:

Tasks (BP-req) = σ (Task-req.Task-reqG == BP-req)(Task-reqs) (Equation 5)

A BP-req is defined by gathering all deconstructed Task-reqs:

BP-req = U Task-req $_{tr}$. (Equation 6)

As a Task requests one or more services to implement its functional objectives, we have to select the most convenient services to implement these Task-requirements. Each functional requirement is broken down into a series of Service Requirement (Ser-req). These service requirements are formalized to select convenient services. A Ser-req is defined as a tuple:

Ser-req$_{sr}$=(Ser-reqN,Ser-reqG,Ser-reqO,Ser-reqNFP,Ser-reqCxt) (Equation 7)

Where

- Ser-reqN defines Ser-req's name;
- Ser-reqG defines this Ser-req's objectives (it also defines which Task-req requests this Ser-req);
- Ser-reqO defines which operations are contained by this Ser-req.
- Ser-reqNFP defines required Non Functional Properties.
- Ser-reqCxt defines this Ser-req's implementation context.
- 'sr' is the Ser-req number.

According to this definition, all Ser-reqs associated to the same Task-req can be gathered by the selection function σ that matches the Task-reqN with Ser-reqG:

Ser-reqs(Task-reqtr)=σ(Ser-req.Ser-reqG==Task-reqN)(Ser-reqs) (Equation 8)

4.2.2 Formalization of Deconstructed Requirement

After deconstructing the management requirement, each service requirement is formalized as a Business Service Requirement (BS-req) by specifying the required Non Functional Property used to constrain service Functional Property and composes additional context information. This formalized BS-req will be used to select convenient services from a massive amount of options.

A BS-req is defined as a tuple:

BS-req$_{bsr}$ = (BS-reqN, BS-reqG, BS-reqFP (Ser_FP, OP_FP), BS-reqNFP (NFP_group (NFP_subgroup, (NFP_CSF, metrics))), BS-reqCxt) (Equation 9)

Where

- BS-reqN defines the BS-requirement's name which can be used to distinguish a BS-req from others.

- BS-reqG defines this BS-req's goal i.e. which Task-req will be implemented by this BS-req;

- BS-reqFP defines required service's Functional Property (FP). This required service Functional Property will be used to select the services depending on the Functional Property.

- BS-reqNFP defines required Non Functional Property for constraining required Functional Property. This BS-reqNFP contains specified Non Functional Property group, sub-group and Critical Success Factor with metrics. These Non Functional Properties are used to select services depending on the Non Functional Property.

- BS-reqCxt defines this BS-requirement's execution condition and other additional information about this BS-requirement.

- 'bsr' is the BS-req's number.

According to this definition we can gather all the BS-req associated to the same Task_req:

BS-reqs (Task-reqtr) = σ (BS-req.BS-reqG == Task-req)(BS-reqS), BS-REQ = U BS-req(Task-reqtr) (Equation 10)

This BS-req's structure is associated to the BS-req's Schema (see Figure 32):

```
<xs:element name="BSRequirement">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="BS-reqN" type="xs:string"/>
      <xs:element name="BS-reqG" type="xs:string"/>
      <xs:element ref="BS-reqFP"/>
      <xs:element ref="NFP-group"/>
      <xs:element ref="BS-reqNFP" minOccurs="0"/>
      <xs:element name="BS-reqCd" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

BS-req's properties

Figure 32 *BS-req's Structure Schema: BS-req's Properties*

4.2.3 Business Resource Selection Process

The BS-reqs that have been generated are published to select the most suitable services. The Service selection process is a two-direction selection:

1. Our SO-MGA is able to select published services according to their provided specifications.

2. Service providers (SPs) are able to orchestrate their services and infrastructures elements to match and apply for these published BS-reqs.

We extract Functional Properties and Non Functional Properties from the specification published by the service providers to formalize these services in a similar format as the BS-reqs.

We define a candidate service as a tuple:

CandidateService $_{cs}$ = (Service_ID, Service_G, Service_FP(OP), Service_NFP, Service_Cxt) (Equation 11)

Where

- Service_ID defines service's identity;
- Service_G defines this Service's objections;
- Service_FP defines this Service's Functional Properties and it also defines what Operations this Service contains;
- Service_NFP defines this Service's Non-functional Properties and their measurement metrics.
- Service_Cxt defines this Service's additional information.
- 'cs' is the number of the Candidate Service.

Candidate services are firstly selected according to their functional properties using the selection function σ which matches Service-FP with BS-reqFP:

FP-satisfiedServices= σ (BS-req.BS-reqFP (Ser_FP, OP_FP) == CandidateService.Service_FP (Ser_FP, OP_FP))(CandidateServices).(Equation 12)

Then a second selection based on non functional properties is done among the services that have matched the functional properties. To this end, we divide each service's Non Functional Properties into two groups: Required Non Functional Property group which matches with BS-REQ required Non Functional Properties and Additional Non Functional Property group:

FP-satisfied.Service_NFP = FP-satisfied.(Service_reqNFP ∪ Service_addNFP). (Equation 13)

The selection function is also used to select that satisfy the Non Functional Property by matching BS-reqNFP with Service_reqNFP:

reqNFP-satisfiedServices= σ (BS-req.BS-reqNFP == FP-satisfiedService.Service_reqNFP)(FP-satisfiedServices). (Equation 14)

After these selections, we design a Comparison Report (COM_REP). This report aims at providing more detailed reference data to support efficiently the service selection from a large number of candidate Services.

The comparison report (COM-REP) includes four major comparison aspects: comparison of functional properties (COM_FP), comparison of required Non Functional Property (COM_ReqNFP), comparison of additional Non Functional Property (COM_AddNFP) and comparison of context (COM_Cxt):

COM_REP = COM_FP ∪ COM_ReqNFP ∪ COM_AddNFP ∪ COM_Cxt. (Equation 15)

A COM_REP contains a set of comparison (Com_rep) between each candidate service and business requirement and comparison between candidates:

Com_rep ∈ COM_REP. (Equation 16)

Each comparison contains the four aspects:

Com_rep=Com_FP∪Com_ReqNFP∪Com_AddNFP∪Com_Cxt (Equation 17)

Where

- Com_FP = Comp (Service.Service_FP, BS-req.BS-reqFP): Compare required Functional Property with Service provided Functional Property;

- Com_ReqNFP= Comp (Service.Service_reqNFP, BS-req.BS-reqNFP): Compare required NFP;

- Com_AddNFP = Comp (Services.Service_addNFP): besides required Non Functional Property if Services provide other Non Functional Properties then compare these additional Non Functional Properties between Services and give detailed reference data to Business Decision Maker;

- Com_Cxt = Comp (Service.Service_Cxt, BS-reqCxt): Compare Service's required context with BS-req's required context.

After confirming Business Decision Maker's selection, the connection between Business Decision Maker and selected Service Providers is created.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="ComparisonReport">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="COM_FP"/>
        <xs:element ref="COM_NFP-Req"/>
        <xs:element ref="COM_NFP-Add"/>
        <xs:element name="COM_Cxt" type="xs:string"/>
      </xs:sequence>
      <xs:attribute name="version" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="COM_FP">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="COM_BP-FP" type="xs:string"/>
        <xs:element name="COM_Ser-FP" type="xs:string"/>
        <xs:element name="COM_OP-FP" type="xs:string"/>
      </xs:sequence>
```

Comparison Report's Properties

Figure 33 *Comparison Report's Schema*

4.2.4 Use Case

The deconstruction and formalization process of Business Decision Maker's management requirement will be illustrated thanks to our Logistics Company's Business Process Use Case. The functional property (FP) of this Logistics process is to distribute the right item in the right quantity at the right time at the right place in the right condition to the right customer. To achieve the non-functional goal, the Logistics Company requires evaluating its business processes by monitoring all involved resources.

This Use Case is achieved using 3 major steps. Each step is completed by several operations and activities (see Figure 34):

- Step 1: formalize Business Decision Maker's Non Functional Property-based management requirement;
- Step 2: select convenient resources.
- Step 3: reach multi-level agreements, prepare governance processes.

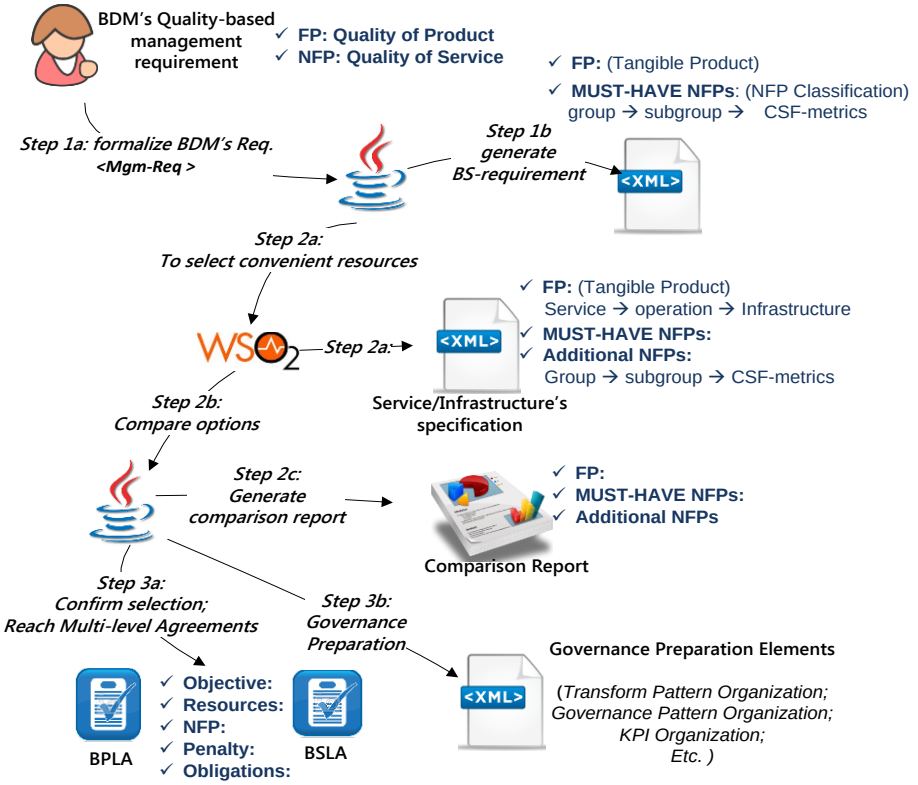


Figure 34 *Organization of BaaS Management and Governance Preparation Use Case*

The Logistics company's business process aims at distributing right goods from factory to right end clients. The entire distribution journey includes

four stops (1. from Factory to Depot; 2. to Distribution Center; 3. to Transmit point; 4. to clients). Depending on the distance and deadline time, there are different transport means that can be used (by air, by train, by road) to distribute goods to next stop. We take each stop as a distribution node. The sub-process from one distribution node to the next node can be defined as a task (or a sub-BP). Therefore, this entire distribution business process is composed of four sub-Business Processes. These sub-Business Processes require different services. Each service can be a black-box service (i.e. can only be controlled at high level, just focus on input and output) or a composed white-box service (i.e. can be fully controlled, from Business Process level to infrastructure level, the internal processes and resources can be modified and configured).

We're showing here how appropriate resources use selection process to build required business scenario according business management requirement. This process includes three steps: (1) Deconstructing abstract business scenario management requirement (Mgm-REQ) to high-level business process requirement (BP-req). (2) Transforming high-level BP-req to precise business service selection requirements (BS-req). (3) Using generated precise BS-reqs to select required services, paying attention on Non Functional Properties. After the selection process, a comparison report is generated for Business Decision Maker to confirm the final decision. Then the required business scenario is built.

Step 1 Deconstruction of Mgm-REQ to BP-req:

The management requirement is defined by:

Mgm-REQ = "build two types of Business Process delivery goods from factory A to client X with minimum cost go through the four distribution nodes. Type one is Express Delivery which means all delivery should be completed in 3 days; Type two is Standard Delivery which means all delivery should be completed during 10 days."

This Mgm-REQ is deconstructed into 2 BP-reqs:

- BP-req 1 = "Express Delivery Business Process with four distribution nodes (Depot, Distribute Center, Transmit Point, Client), within 3 days";
- BP-req 2 = "Standard Delivery Business Process with four same distribution nodes within 10 days".

Each BP-req is deconstructed into four Tasks:

- Task 1: Distribute goods from factory to Depot;
- Task 2: Distribute goods from Depot to Distribute Center;
- Task 3: Distribute goods from Distribute Center to Transmit Point;
- Task 4: Distribute goods from Transmit Point to end-clients.

Step 2 Formalized high-level BP-req to precise BS-req:

According to the deconstructed Task, for BP-req 1, four precise BS-reqs are generated:

BS-req 1-1= ("BS-req 1-1", "distribution: from factory to depot", "distribution service (FP: "inventory", "shipping")", "cost ("price", metrics); delivery time ("Express")", "BS-Cxt")

BS-req 1-2= ("BS-req 1-2", "distribution: from depot to distribution center", "distribution service (FP: "inventory", "Express shipping")", "cost ("price", metrics); delivery time ("Express")", "BS-Cxt")

BS-req 1-3= ("BS-req 1-3", "distribution: from distribution center to transmit point", "distribution service (FP: "inventory", "Express shipping")", "cost ("price", metrics); delivery time ("Express")", "BS-Cxt")

BS-req 1-4= ("BS-req 1-4", "distribution: from transmit point to client", "distribution service (FP: "inventory", "Express shipping")", "cost ("price", metrics); delivery time ("Express")", "BS-Cxt")

For BP-req 2, following four precise BS-reqs are also generated:

BS-req 2-1= ("BS-req 2-1", "distribution: from factory to depot", "distribution service (FP: "inventory", "shipping")", "cost ("price", metrics); delivery time ("Standard")", "BS-Cxt")

BS-req 2-2= ("BS-req 2-2", "distribution: from depot to distribution center", "distribution service (FP: "inventory", "Express shipping")", "cost ("price", metrics); delivery time ("Standard")", "BS-Cxt")

BS-req 2-3= ("BS-req 2-3", "distribution: from distribution center to transmit point", "distribution service (FP: "inventory", "Express shipping")", "cost ("price", metrics); delivery time ("Standard")", "BS-Cxt")

BS-req 2-4= ("BS-req 2-4", "distribution: from transmit point to client", "distribution service (FP: "inventory", "Express shipping")", "cost ("price", metrics); delivery time ("Standard")", "BS-Cxt")

Step 3 we use the formalized BS-reqs to select convenient services, and to build required business scenario:

After selecting convenient services, a comparison report is generated for Business Decision Maker to confirm the selection. Then after the confirmation, the selected services are organized to build the required business scenario. For example, there are two services (a. Express Delivery Service from Depot to Distribution Center, b. Standard Delivery Service from Depot to Distribution Center) that are selected according to our formalized BS-reqs (see Figure 35):

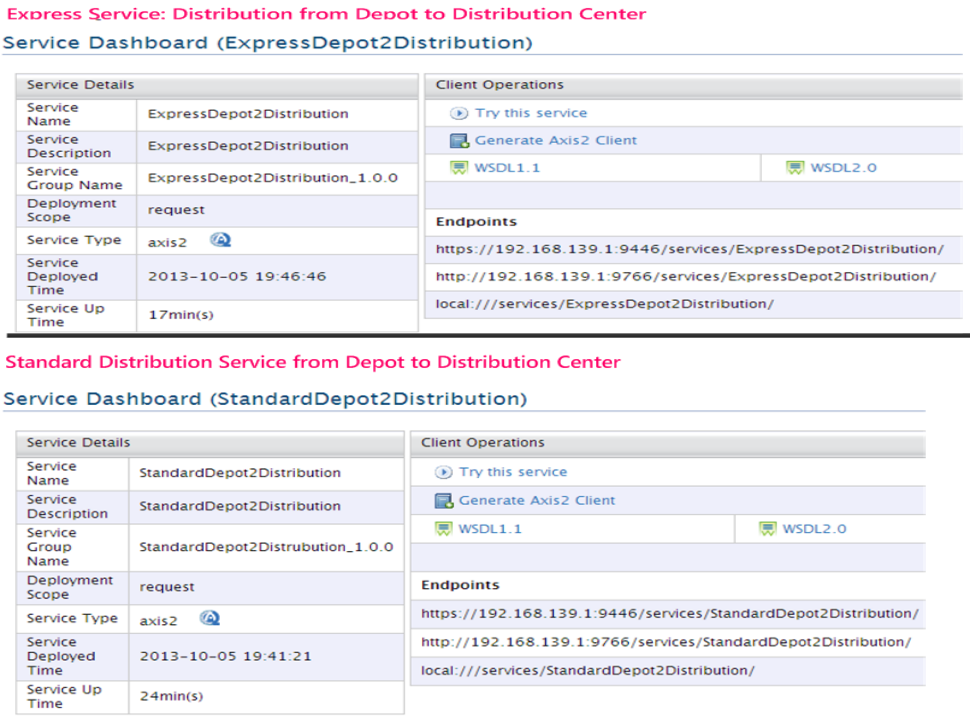


Figure 35 *For Example: Selected Services*

In this use case, the required business scenario is built thanks to two types of Delivery Business Process with seven Services (See Figure 36): (1) Express Delivery Business Process is implemented by Service 1, Service E2, Service E3, and Service E4; whereas (2) Standard Delivery Business Process is implemented by Service 1, Service S2, Service S3, and Service S4. Service 1 is used for both Express Delivery and Standard Delivery.

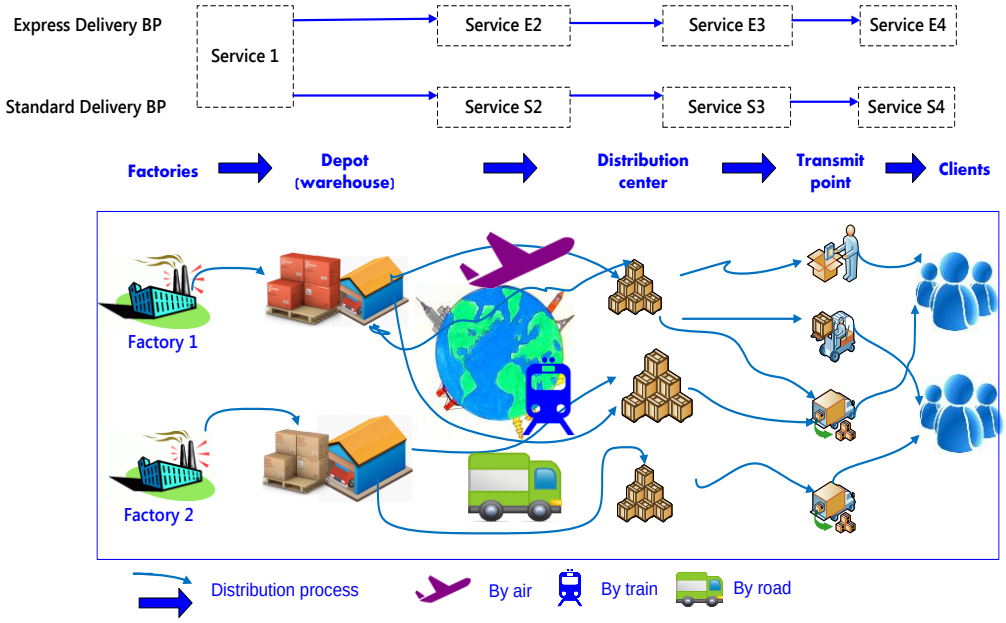


Figure 36 *Use Case - Business Scenario*

4.3 Negotiation and Governance Preparation Model

The Negotiation and Governance Preparation Model aims at simplifying the multi-level agreements building process and preparing governance. As stated section 3.2, this model drives Business Resource Manager and Pre-Governance Manager. Figure 37 presents the interaction and information exchange. This model's elements are organized as follow (see Figure 38):

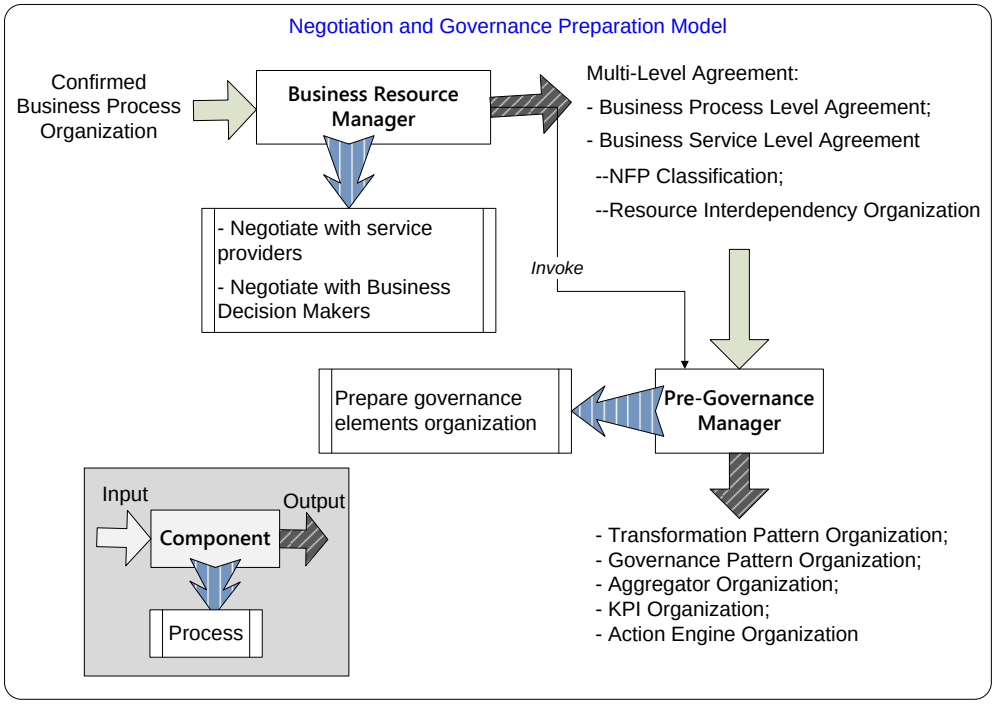


Figure 37 *Interaction of Negotiation and Governance Preparation Components*

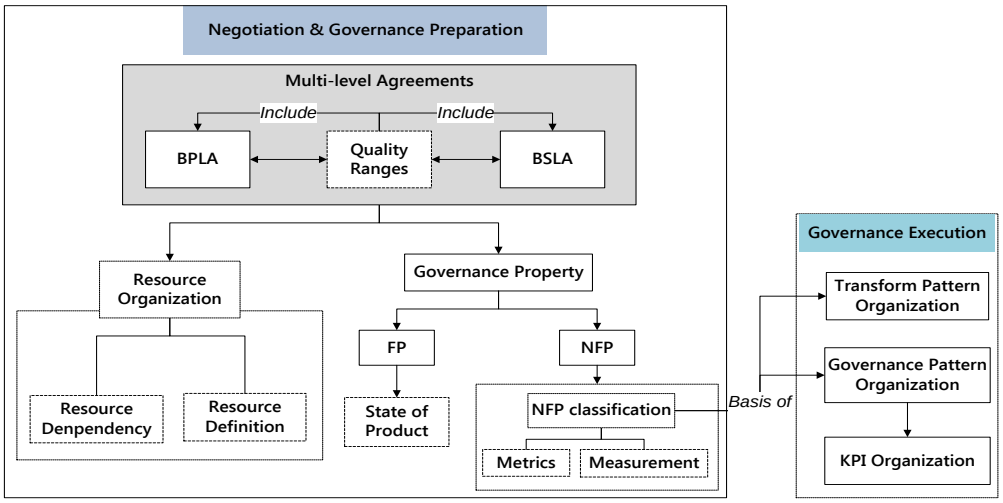


Figure 38 *Negotiation and Governance Preparation Model*

BPLA and BSLA have a similar structure. They define the obligations and penalties of participants. They also define the Non Functional Property's organization and Quality Ranges. These Multi-level Agreements' Schema and resource definition's schema are shown Figure 39.

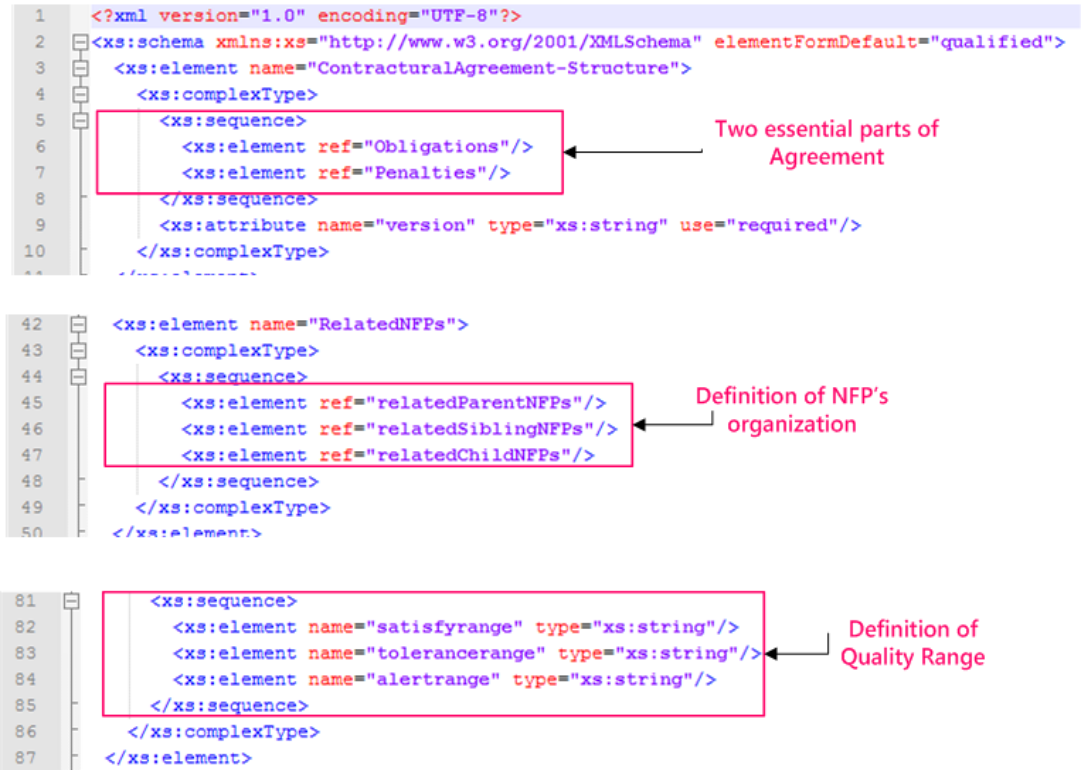


Figure 39 *Excerpts of Multi-level Agreements*

4.3.1 Business Process Level Agreement

BPLA is the agreement set in this Business Process Layer. It pays attention to four types of resources: Business Process, Task, Service and Operation. A BPLA is composed of two parts: Obligations and Penalties. Obligations part defines required resources' Functional Properties, Non Functional Properties and other properties; Penalties part defines punitive processes. Once Service Oriented Management Governance Architecture provided BaaS cannot meet its promised quality or if any BPLA violation occurs, these punitive processes will be invoked.

In this section we focus on the definition of resources. Formally, a Business Process is defined as a tuple, it is deployed in Business Process Layer:

BP_{bp}= (BP_ID, BP_G, BP_FP (Tasks), BP_NFP (NFPs, Metrics), BP_Cxt, BP_Ep), (Equation 18)

Where

- BP_ID defines Business Process' identity;

- BP_G defines the objectives of this Business Process;

- BP_FP (Tasks) defines the functional properties of this Business Process, and it also defines which Tasks will be requested to implement this Business Process.

- BP_NFP defines non-functional properties of this Business Process and the metrics to measure these Non Functional Properties.

- BP_Cxt defines the necessary context of this Business Process' implementation;

- BP_Ep defines this Business Process' endpoint which is the entry of this Business Process.

- 'bp' is the Business Process' serial number.

All Business Processes can be gathered together: $\cup BP_{bp}$.

A Task is defined as a tuple, it is deployed in Business Process Layer:

Task $_t$ = (Task_ID, Task_G, Task_FP(Services), Task_NFP(NFPs, Metrics), Task_Cxt, Task_Ep),

(Equation 19)

Where

- Task_ID defines the identity of task;

- Task_G defines the goal of this task. It also defines which Business Process re-quires this Task;

- Task_FP (Services) defines the functional properties of this Task. It also defines which services will be requested by this task;

- Task_NFP (NFPs, Metrics) defines non-functional properties of this Task and the metrics to measure these NFPs.

- Task_Cxt defines the necessary context of this Task's implementation;

- Task_Ep defines the endpoint of this task which is the entry to this task and its description;

- 't' is the Task's serial number.

According to this definition, Tasks attached to any BP_{bp} can be identified by selecting (σ) the Tasks while the Task_G matches with BP_{bp}'s identity:

Tasks (BP_{bp}) = σ (Task.Task_G == BP_ID)(Tasks). (Equation 20)

Based on the definition of candidate Services, in BPLA a Service is also defined as a tuple in the similar way:

Service $_s$ = (Service_ID, Service_G, Service_Type, Service_FP(OPs), Service_NFP(NFPs, Metrics), Service_Cxt, Service_Ep), (Equation 21)

Where

- Service_ID defines service's identity;

- Service_G defines objectives of this service. It also defines which Task requires this Service;

- Service_Type defines service's type: black-box or white-box. For a black-box service we only need to pay attention on its inputs and outputs. For a white-box service, its internal processes and required lower level resources can be configured and fully controlled.

- Service_FP(OPs) defines functional properties of this Service. It also defines which Operations will be requested by this Service;

- Service_NFP(NFPs, Metrics) defines specified Non Functional Properties of this Service and metrics;

- Service_Cxt defines the necessary context of this Service's implementation;

- Service_Ep defines the endpoint of this Service.

-'s' is the serial number of the Service.

According to the definition, all the Services associated to the $Task_t$ can be gathered using the selection function σ. The selection function matches Service's objective (Service_G) with Task's identity (Task_ID):

Sevices(Task_ID)=σ(Service.Service_G==Task.Task_ID)(Services) (Equation 22)

An Operation is used to support the associated Service's execution by operating certain infrastructure elements. To keep the consistency, an Operation is also defined as a tuple:

OP_o = (OP_ID, OP_G, OP_Type, OP_FP (infrastructure), OP_NFP(NFPs, Metrics), OP_Cxt, OP_Ep) (Equation 23)

Where

- OP_ID defines the identity of this operation;

- OP_G defines the goal of this operation. It also defines which Service requires this Operation;

- OP_Type defines the type of this operation, whether it is a black-box operation which only is focused on its input and output, or it is a white-box operation which its internal process and required lower level resources are able to be configured and fully controlled.

- OP_FP defines which infrastructure elements this operation requests;

- OP_NFP defines Operation's Non Functional Properties and Metrics;

- OP_Cxt defines the necessary context of this Operation's implementation;

- OP_Ep defines the endpoint for this operation;

- "o" is the operation's serial number.

In a similar way, we can gather a set of operations associated to the same services thanks to the selection function by matching the Operation's objective (OP_G) with a Service's name (Service_ID):

OPs (Services) = σ (OP.OP_G == Services.Service_ID)(OPs). (Equation 24)

To manage and control quality of business activities Non Functional Properties, we divided the Quality Range of Business Process (QoBP) into 3 sub-ranges:

QR(BPLA)=Satisfied Range ∪ Tolerable Range ∪ Alert Range. (Equation 25)

- Satisfied Range (SR): means that the reached quality value highly fits the Non Functional Property requirement;

- Tolerable Range (TR): means that the quality value barely meet the Non Functional Property requirement. If this quality level cannot be improved, it is likely to violate BPLA. Therefore, the governance system should launch Action Processes to re-configure Business Service layer and Business Implementation layer to improve this quality level.

- Alert Range (AR): means that the quality value cannot satisfy the Non Functional Property requirement. Business Decision Maker is notified and governance system is facing a penalty.

These sub-ranges are separated by two types of Threshold:

- Threshold (ThS-T) is used to divide Satisfied Range and Tolerable Range;

- Threshold (ThT-A) is used to divide Tolerance Range and Alert Range;

The values of these thresholds can be negotiated between Business Decision Maker and SO-MGA when the BPLA is created. Metrics include the values of these two thresholds.

For example, in our Use Case the business scenario is built in section 4.2.4, Non Functional Property of Service1 can be defined as: Service_NFP = (NFP="the response delay rate in an hour"; NFP.metrics: "Threshold(ThS-T) = 1%; Threshold (Th $_{T-A}$) = 5%") this means if this Service_NFP < 1%, it is in satisfactory range; if ("1 %"> Service_NFP > "5%") then the QoS on this Non Functional Property is in the Tolerable Range, the Service Oriented Management Governance Architecture should launch Action Processes to negotiate with this Service's Provider, try to improve the QoS on this Non Functional Property. If the Action Processes achieved the improvement, made this Service_NFP in the Satisfied Range, for SO-MGA, the BPLA reconfiguration and penalties were avoided, for Business Decision Makers their Non Functional Properties requirements have been satisfied. Otherwise, once Service_NFP > 5%, it is in Alter Range, the defined punitive processes will be invoked.

4.3.2 Business Service Level Agreement

As we said previously, Business Service Level Agreement (BSLA) names the agreement in Business Service Layer (BSL) and Business Implementation Layer (BIL). It is a technical version of Business Process Level Agreement (BPLA).

BSLA pays attention to three resources: Service, Operation and Infrastructure elements. In the similar way with previously introduced BPLA, BSLA is also composed of two parts: Obligations and Penalties. Obligations part defines required resources' Functional Properties, Non Functional Properties and other properties. Penalties part defines punitive processes. Once any provided Services cannot meet the promised quality or if there is any BSLA violation occurred, these punitive processes will be invoked against the service provider.

BSLA's Service and Operation are defined in a similar way as the Service and Operation in BPLA. Nevertheless, this Business Service Level Agreement uses a more technical language to describe elements in Service and Operation definition tuples. In order to keep consistency we also define an Infrastructure element as a tuple:

$Infrastructure_{inf}$= (Inf_ID, Inf_G, Inf_FP, Inf_NFP(NFPs, Metrics), Inf_Cxt), (Equation 26)

Where

- Inf_ID defines infrastructure element identity,
- Inf_G defines infrastructure element objectives,
- Inf_FP defines infrastructure element Functional Property. It also defines which Operation re-quires this Infrastructure,
- Inf_NFP defines Non Functional Property and Metrics used to constrain infrastructure's Functional Property,
- Inf_Cxt defines infrastructure element necessary context. It is used to describe customized additional information.
- 'inf' is the serial number Infrastructures.

According to this definition, a set of Infrastructure element for any Operation (OP_o) can be gathered by invoking the selection function σ:

Infrastructures (OP_o) = σ (Infrastructure.Inf_FP == OPo.OP_N)(Infrastructures). (Equation 27)

In order to minimize the impact of Business Service Level Agreement (BSLA)'s violation to quality of business process, we define the value of Quality Range for BSLA. This Quality Ranges (QRs) are defined using the same formulation as for the quality ranges in Business Process Level Agreement (BPLA).

In order to hind the violation of Business Service Level Agreement (BSLA) from business process perspective, BSLA should have lower tolerance for quality of resource in business services layer and business implementatyion layer. Thus, the lower tolerance can warn the governance system to react the violation before the violation harms business process performance.

We continue to take the Service1's response delay rate as an example. In Business Service Level Agreement, Service1's quality ranges are presented as follow:

Response Delay Rate	BPLA	BSLA
Th $_{S-T}$	1%	0.7% (<1%)
Th $_{T-A}$	5%	3% (<5%)

4.3.3 Classification of Non Functional Property and Resource Dependency

After setting Multi-level Agreements, the Governance Preparation process is activated to define and organize the required governance elements. We define six essential governance elements: 1) Classification of Non Functional Property, 2) Resource Dependency, 3) Transformation Pattern, 4) Governance Pattern, 5) Key Performance Indicator and 6) Aggregator. The interactions between these elements will be introduced in detail in the next chapter (chapter 5 Governance Execution and Adapting).

Classification of Non Functional Properties is the primary element of our governance process. As mentioned in section 3.3, Non Functional Property is a critical property of a Resource. Non Functional Property constrains the way Resource's Functional Property is achieved. To govern performance of resources we have to pay attention on the Non Functional Property quality. To support tuned governance, we classify Non Functional Properties into different groups and divide each group into precise sub-group accordingly. Each sub-group is constrained by one or more Critical Success Factor. Classifying Non Functional Properties aims at providing different abstraction levels on Non Functional Properties, from a global vision (for the NFP groups) to precise Critical Success Factor selection. By this way a precise governance operation process can be set as this classification helps refining high-level requirements to identify convenient KPI to deploy. This approach leads us to organize other essential governance elements such as Transformation Pattern, Governance Pattern and Key Performance Indicator according to this Non Functional Property classification.

XML is used to transport and store data in our governance process. All required Non Functional Properties in Multi-level Agreements are organized in a file < NFP Classification Organization.xml > (see Figure 40). A Non Functional Property is defined as a tuple. Its properties contain its value and the relationships with other Non Functional Properties.

NFP $_{num}$ = (Name, Elements, Measurement, AggregationAlgorithm, {ChildNFP}, {ParentNFP}) (Equation 28)

Where

- NFP-Name defines the Non Functional Property name;

- NFP-Element defines this Non Functional Property's data structure elements;

- NFP-Measurement defines this Non Functional Property's monitoring measurement;

- NFP-Algorithm defines this Non Functional Property's aggregating algorithm;

- NFP-ParentNFP defines this Non Functional Property's associated upper level's Non Functional Property (i.e. for Delay Rate, its parentNFP is the Performance Group).

- NFP-ChildNFP defines this Non Functional Property associated lower level's Non Functional Property. This means that, this Non Functional Property's performance depends on all of the ChildNFPs' performance (i.e. a Group-NFP's related CSF-NFPs are children-NFP for this Group-NFP and they will be composed to reflect the performance of their ParentNFP).

- "num" is the number of this Non Functional Property.


```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="NFPclassification">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="NFP" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="version" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="NFP">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="name" type="xs:string"/>
        <xs:element ref="Elements"/>
        <xs:element name="Measurement" type="xs:string"/>
        <xs:element ref="Algorithm"/>
        <xs:element ref="ParenetNFP" minOccurs="0"/>
        <xs:element name="ChildNFP" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="Elements">
  <xs:element name="Algorithm">
  <xs:element name="AggregationAlgorithem">
  <xs:element name="ParenetNFP">
</xs:schema>
```

Figure 40 *The Schema of NFP's Classification*

As explained previously, the information contained in this classification file is exchanged to achieve Non Functional Property's governance by extracting all related children Non Functional Properties and their Critical Success Factors.

After building this Non Functional Property Classification, Transformation Pattern, Governance Pattern and Key Performance Indicator are organized according to this Non Functional Property Classification.

In addition, our governance process takes into account the business process organization. This allows taking advantage of the Resource organization to organize governance process. Therefore, Resource's dependence relationships are organized and stored in a file < ResourceDependency.xml >. This file contains all required Resources' type, layer, and their relationship with other Resources.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="ResourceTypeRepoitory">
  <xs:element name="resourceType">                    ResourceType:
    <xs:complexType>                                  { BP/Task/Service/Operation/Infrastructure }
      <xs:sequence>
        <xs:element name="Layer" type="xs:string"/>          Layer:
        <xs:element ref="resources" minOccurs="0"/>          {BPL/BSL/BIL}
        <xs:element ref="resource" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="resources">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="resource" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="resource">                        Resource:
    <xs:complexType>                                  Resource ID
      <xs:sequence>
        <xs:element name="resourceID" type="xs:string"/>
        <xs:element ref="require" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="require">                         Require :
    <xs:complexType>                                  required resource
      <xs:sequence>
        <xs:element name="requireID" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Figure 41 *Schema of Resource Dependency*

This Resource Dependency file (generated after reaching Multi-level Agreement) stores the dependency information of required Resources. According to the Resource Dependency information, Resources can be extracted by a simple parsing process while organizing the customized governance according to a particular Business Service.

4.3.4 Definition of Transformation Pattern and Governance Pattern

Transformation Pattern (Pat) is an important governance element. These patterns are organized according to the Non Functional Property classification. They aim at transforming Monitoring Requirements into Monitoring Policy Rules by specifying governed Non Functional Property and resource. Transformation Patterns are defined and organized in Governance Preparation Phase. All registered Transformation Patterns are recorded in a file <Transformation Pattern.xml > (see Figure 42). Monitoring Requirement can be transformed to Monitoring Policy Rules by parsing this Transformation Pattern file and considering Resource Dependency.

A transformation pattern is defined as a tuple:

Pat_j = (PatN, PatG,{PatCxt},PatP,{PatCol},{PatR},{PatCsq})

(Equation 29)

Where

- PatN identifies the pattern. This pattern name is related to the requirement type, a Non Functional Property identification or a Critical Success Factor for a group of Non Functional Property.

- PatG defines the pattern goal (or the reason for using it). This goal is similar to the requirement goal and the associated value can be either governance or any other functional requirement related to a Non Functional Property or Critical Success Factor.

- PatCxt identifies the context under which this pattern can be used.

- PatP defines a list of “participants” and their roles in the pattern definition. A participant can refer to a requirement which need to be transformed, transformed policy rule, relevant sub-pattern, collaborative business process organization, etc.

- PatCol describes the collaboration strategy used by the participants to interact with each other.

- PatR a set of related patterns. For example, governance requirement pattern requires a set of the Critical Success Factor pattern for a parent NFP group pattern.)

- PatCsq describes the results or actions implemented by the pattern.

In a similar way, the set of patterns is defined by:

$Pats = \cup Pat_j$, (Equation 30)

- “j” is the pattern number.

Figure 42 *Excerpt of Transformation Pattern's Definition Schema*

The Governance Patterns are designed to implement Real-time Monitoring Policy Rules by invoking Key Performance Indicators and transporting all necessary information from Monitoring Policy Rules to Key Performance Indicators. Governance Patterns are also defined according to the Non Functional Property Classification and to the Resource Dependency Organization. Registered Governance Patterns are stored in a file <Governance Pattern.xml> file (see Figure 43). Convenient Governance Patterns can be extracted by parsing this file according to the required Governance Pattern name. Extracted Governance Patterns can invoke related Key Performance Indicators to implement Monitoring Policy Rules.

A Governance Pattern gp is defined as a tuple:

$GPat_{gp}$= (GPat-ID, GPat-Goal, {GPat-Cxt}, GPat-KPI, GPat-Trans ((PolR-resource, KPI-resource), (PolR-Gov, KPI-Gov)), {GPat-Csq})

(Equation 31)

Where

- GPat-ID defines Governance Pattern's identity;
- GPat-Goal defines the objectives of this Governance Pattern;
- GPat-Cxt defines the context of this Governance Pattern's implementation;
- GPat-KPI defines the deployed Key Performance Indicator. A Monitoring Policy Rule's information will be transported to this Key Performance Indicator to implement the monitoring policy rule.
- GPat-Trans ((PolR-resource, KPI-resource), (PolR-Gov, KPI-Gov)) take Policy Rule's resource information and Non Functional Property's Critical Success Factor information as Key Performance Indicator's input, transport to invoked Key Performance Indicator.
- GPat-Csq defines the consequences of this Governance Pattern's implementation.

A set of Governance Pattern is defined by:

$GPats = \cup GPat_{gp}$, (Equation 32)

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="GovernancePatterns">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="GovernancePattern" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="version" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="GovernancePattern">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="GPat-ID" type="xs:string"/>
        <xs:element name="GPat-Goal" type="xs:string"/>
        <xs:element name="GPat-Context" type="xs:string"/>
        <xs:element name="GPat-KPI" type="xs:string"/>
        <xs:element ref="GPat-Trans"/>
        <xs:element name="GPat-Csq" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="GPat-Trans">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="PolR-resource" type="xs:string"/>
        <xs:element name="PolR-Gov" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

GPat's Properties

GPat's transport info

Figure 43 *Schema of Governance Pattern*

4.3.5 Definition of Key Performance Indicators and Aggregator

Key Performance Indicators (KPIs) are designed to monitor the performance and quality of a Resource's precise Critical Success Factor at run time. Besides measurement and monitoring, the accuracy of governance can be improved. Key Performance Indicator is orchestrated as an alert mean. Once an unexpected situation is detected, the Key Performance Indicator management component will send an alarm signal to the administrator immediately, to make execution system be able to respond quickly to unexpected situations. Registered Key Performance Indicators' properties are stored in a file <KPI Repository.xml > (see Figure 44). By parsing this file Key Performance Indicators information can be exchanged during the governance process, and convenient Key Performance Indicators can be invoked by selecting their properties.

A Key Performance Indicator is defined as a tuple:

KPI_y= (KPI-ID, KPI-Goal, KPI-Resource (name, type, layer), KPI-Gov (CSF/ CSF-Pattern), KPI-Output (distance, data, Timestamp), KPI-Csq)

(Equation 33)

Where

- KPI_ID defines Key Performance Indicator's identity to distinguish a Key Performance Indicator from others;
- KPI-Goal defines Key Performance Indicator's objectives;
- KPI-Resource (name, type, and layer) defines all essential information of Key Performance Indicator's target resource;
- KPI-Gov (CSF/CSF-Pattern) defines target Non Functional Property's Critical Success Factor information. It can track back to Critical Success Factor's metrics and measurement;
- KPI-Output (distance, data, Timestamp) defines Key Performance Indicator's monitoring results. The "distance" is defined as distance=D[0,1]. If the timely recording matches the given metric of this monitoring execution, then "distance=D[0]". In such case, the execution of the policy rule is "normal". Otherwise, if the timely recording does not match the given metric, then the distance=D[1] to the execution policy rule is defined as abnormal. The "data" is associated the timely monitoring record which can be selected by convenient aggregators according computing requirements. Timestamp defines the specific time this data is recorded. It can be used to distinguish a records from others.
- KPI-Csq defines the consequences of this Key Performance Indicator's execution.

According to this definition, we can gather all the Key Performance Indicators as:

$KPIs = \cup KPI_y$, (Equation 34)

- “y” is the Key Performance Indicator’s number.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="KPIrepository">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="KPI" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="version" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="KPI">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="KPI-ID" type="xs:string"/>
        <xs:element name="KPI-Goal" type="xs:string"/>
        <xs:element name="KPI-Resource" type="xs:string"/>
        <xs:element name="KPI-Gov" type="xs:string"/>
        <xs:element name="KPI-Output" type="xs:string"/>
        <xs:element name="KPI-Csq" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Figure 44 *Schema of KPI*

Our Governance Process not only implements real-time monitoring. It also pays attention on computing and aggregating real-time results to comprehensive results. An Aggregator aims at using appropriate algorithms to compute considered Critical Success Factor from the collected real-time results. Aggregating Algorithms depend on Critical Success Factor and Resource’s Composition Type. Therefore, each Aggregator has an Algorithm for particular Critical Success Factor and Resource Composition. This computing and aggregating process is implemented after implementing real-time monitoring process.

Registered Aggregators are stored in a <Aggregator-Algorithms.xml> file (see Figure 45). Convenient Aggregator can be invoked by the Governance Pattern. An Aggregator is also defined as a tuple:

$$Aggregator_{num} = (Agg\text{-}ID, CompositionType, CSF, Algorithm) \quad \text{(Equation 35)}$$

Where

- Agg-ID defines the aggregator identity;
- CompositionType defines which type of Resource Composition suite this Aggregator Algorithm.
- CSF defines which Critical Success Factor can be computed by this Aggregator;

- Algorithm defines this required algorithm for this Aggregator according to particular Critical Success Factor and CompositionType;
- "num" is the number of Aggregator.

The set of Aggregators is defined as:

$$Aggregators = \cup Aggregator_{num} \quad \text{(Equation 36)}$$

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="Aggregators">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="Aggregator" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="Aggregator">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Aggregator-ID" type="xs:string"/>
        <xs:element name="CompositionType" type="xs:string"/>
        <xs:element name="CSF" type="xs:string"/>
        <xs:element name="Aglorithm" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Figure 45 *Schema of Aggregator*

The preparation phase aims at defining some essential governance elements. All the associated files are created for preparing following governance execution. The interaction between these elements and information exchange between these files will be introduced in the next chapter.

4.3.6 Use Case - BaaS Management and Governance Preparation

Running our logistics Use Case will present the implementation of our Negotation and Governance Preparation process. This process is composed of three parts: (1) Resource Definition and Dependencies (2) Non Functional Property Classification, Definition of Transformation Pattern and Governance Pattern; (3) Definition of Key Performance Indicator and Aggregator-Algorithm.

4.3.6.1 Definition and Dependencies of Resources

Our governance solution aims at ensuring the information value chain associated to a business processes fitting the business goals. According to our BaaS Management and Governance Preparation Framework, the governance

process should pay attention on the organization of business resources and extend traditional technical governance to this business perspective. To this end, the first step consists in understanding the organization and interdependencies of business resources.

According to the business scenario which we have built in section 4.2.5, there are two types of Delivery Business Process (a, Express Delivery Business Process, b. Standard Delivery Business Process). We will further focus on the Express Delivery Business Process' business resources' definition and dependencies. Details for the Standard Delivery Business Process are given in Appendix chapter 1.

The formalized definition of Express Delivery Business Process' resources is as following:

BP $_{express}$ = ("Express Delivery BP", "Express: Deliver goods from factory to clients", "Task 1, Task 2, Task 3, Task 4", "Delivery time: < 3 days, Cost: Low", "BP_Cxt", "BP_Ep")

This Business Process is divided into four Tasks accordingly:

- Task 1: Distribute goods from factory to Depot;
- Task 2: Distribute goods from Depot to Distribute Center;
- Task 3: Distribute goods from Distribute Center to Transmit Point;
- Task 4: Distribute goods from Transmit Point to end-clients.

The formalized definition of these Tasks are listed below:

Task 1 = ("Task 1", "distribution from factory to depot", "Service 1", "Delivery time, Cost", "Task_Cxt", "Task_Ep");

Task 2 = ("Task 2", "distribution from depot to distribution center", "Express: Service E2; Standard: S2", "Delivery time, Cost", "Task_Cxt", "Task_Ep");

Task 3 = ("Task 3", "distribution from distribution center to transmit point", "Express: Service E3; Standard: S3", "Delivery time, Cost", "Task_Cxt", "Task_Ep");

(Ep. 1)

Task 4 = ("Task 4", "distribution from transmit point to client", "Express: Service E4; Standard: S4", "Delivery time, Cost", "Task_Cxt", "Task_Ep");

Each Task requires at least one Service. Here after are four required Services:

Service 1 = ("Service 1", "distribution from factory to depot", "black-box", "FP: shipping", "NFP: delivery time; cost", "", "Service1_Ep")

Service E2 = ("Service E2", "distribution from depot to distribution center", "black-box", "FP: shipping", "NFP: delivery time; cost", "", "ServiceE2_Ep")

Service E3 = ("Service E3", "distribution from distribution center to transmit point", "white-box", "FP: shipping (OP E3-1; OP E3-2)", "NFP: delivery time; cost", "", "ServiceE3_Ep")

Service E4 = ("Service E4", "distribution from transmit point to client", "white-box", "FP: shipping (OP E4-1; OP E4-2)", "NFP: delivery time; cost", "", "ServiceE4_Ep")

Each Service is implemented by at least one Operation:

Operation E3-1 = ("OP E3-1", "manage inventory", "black-box", "FP: inventory management", "NFP", "", "OPE3-1_Ep");

Operation E3-2 = ("OP E3-2", "manage shipment", "black-box", "FP: shipment management", "NFP", "", "OPE3-2_Ep");

Operation E4-1 = ("OP E4-1", "manage inventory", "black-box", "FP: inventory management", "NFP", "", "OPE4-1_Ep");

Operation E4-2 = ("OP E4-2", "manage shipment", "white-box", "FP: shipment management (Require: Inf E4-2)", "NFP", "", "OPE4-2_Ep");

For a white-box Operation, the required Infrastructure elements can be managed. In this case for Operation E4-2 the required Infrastructure element E4-2 can be managed.

Infrastructure E4-2 = ("Inf E4-2", "implement OP E4-2", "shipping")

According to the formalization of these Resources, we map these Resources into our Multi-Layer Management and Governance Architecture to show their dependency relationships (see Figure 46)

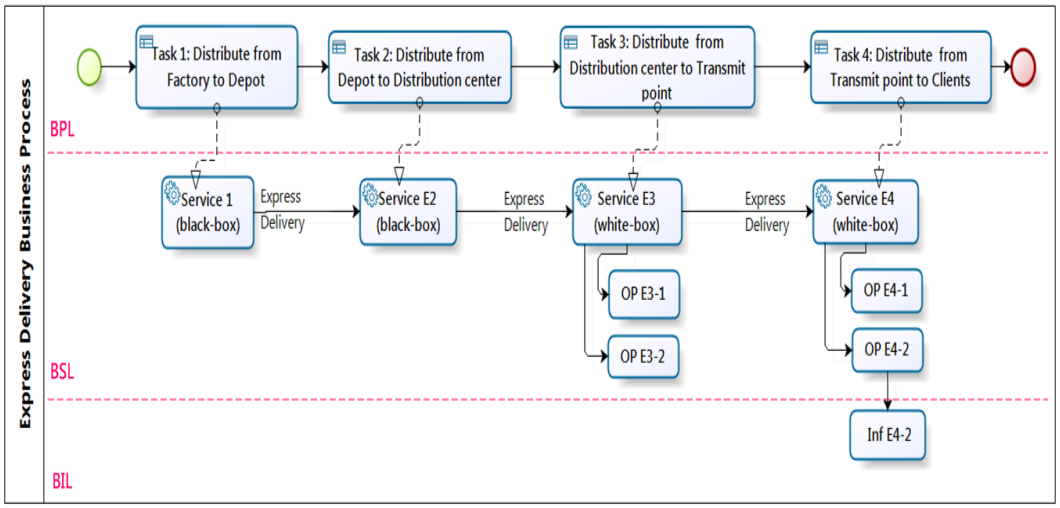


Figure 46 *Express Delivery BP's Resource Dependency*

4.3.6.2 Non Functional Property Classification, Transformation Pattern and Governance Pattern

After identifying the resources we have to refine Non Functional Properties to precise governance requirements. As introduced section 3.3, we classify Non Functional Property into different groups. Each group can be divided into detailed sub-groups. Each sub-group can be divided into Critical Success Factors. These classified and registered Non Functional Properties are critical factors which impact the organization of the proposed governance solution. In our Use Case, there are 23 registered Non functional Properties. We split them into four Groups (Group-Cost; Group-Performance; Group-Maintainability; Group-Security) (see Figure 47). More details about the registry are given in Appendix chapter 2:

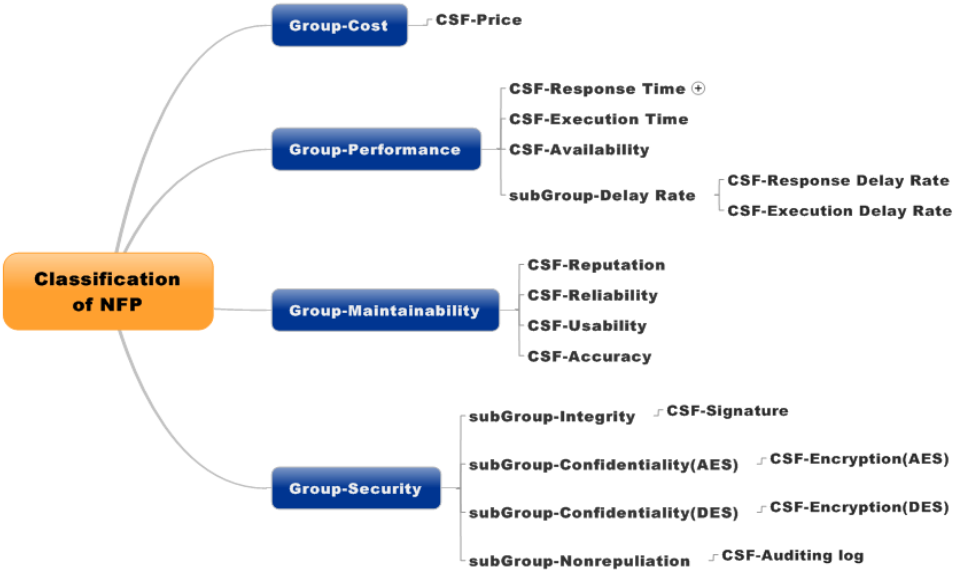


Figure 47 *Use Case - Classification of Non Functional Property*

This Non Functional Property Classification is taken as a basis for organizing our Use Case, Transformation Pattern, Governance Pattern, Key Performance Indicator, etc. So after building Non Functional Property Classification, 23 Transformation Patterns are created. The organization of Transformation Pattern (Pat) is similar with the Non Functional Property classification. A Group-NFP is corresponding to a Pattern. A subgroup-NFP is corresponding to sub-Pattern. Critical Success Factor is corresponding to CSF-Pattern.

Following figure shows the Use Case Transformation Patterns organized in four Groups: Pattern-Cost, Pattern-Performance, Pattern-Maintainability and Pattern-Security. These four Patterns are associated to Non

Functional Property Groups (See Figure 48). More details are given in Appendix chapter 2.

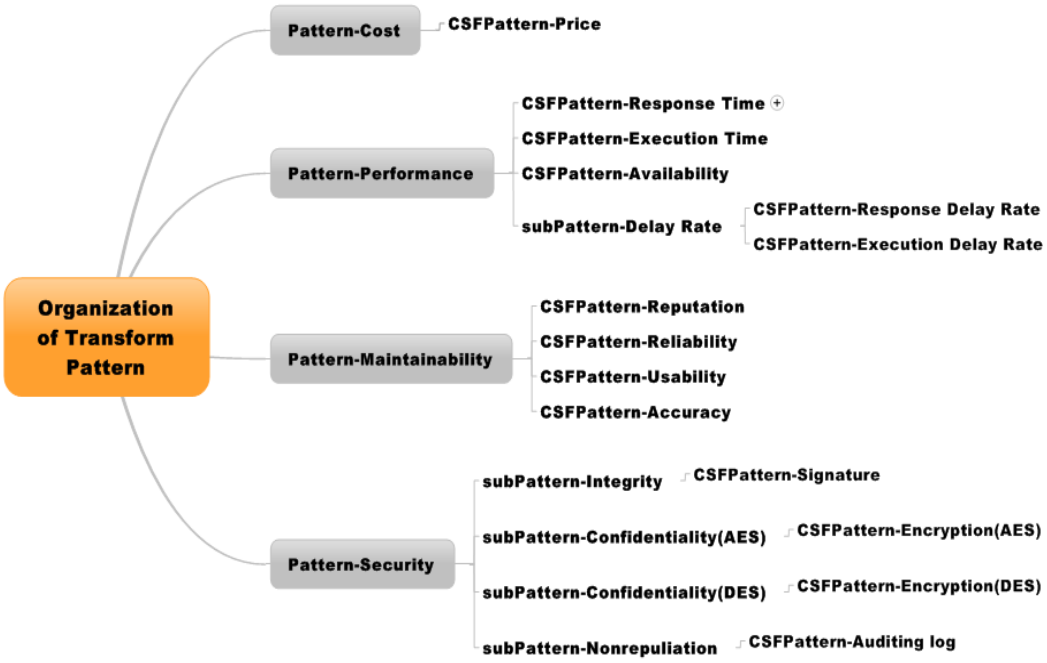


Figure 48 *Use Case - Organization of Transformation Pattern*

Then according to the Non Functional Property classification we derive the Critical Success Factor associated to our Use Case Governance (see Figure 49):

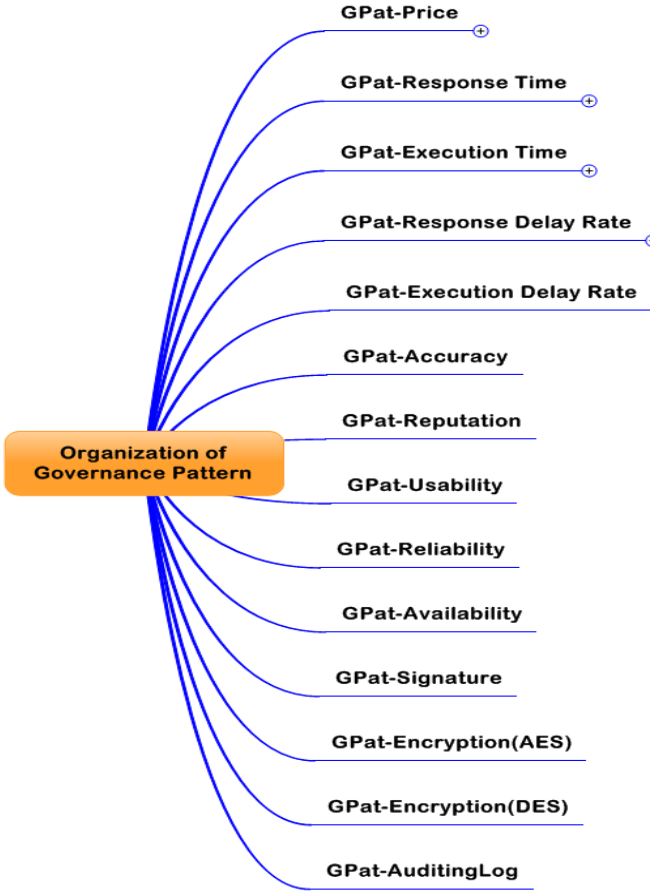


Figure 49 *Organization of Governance Pattern*

After organizing the Governance Pattern, convenient Key Performance Indicators can be invoked by Governance Patterns (GPat) to match these patterns property GPat-KPI with Key Performance Indicator's name. Governance Patterns can be used to transform Monitoring Policy Rule's information to required Key Performance Indicators.

4.3.6.3 Key Performance Indicator, Aggregator-Algorithm

In our Use Case Key Performance Indicators registered in <KPI repository.xml> can be invoked by Governance Pattern. To this end the Key Performance Indicator file is parsed to find convenient Key Performance Indicator, the selection is achieved according to the definition of Key Performance Indicators. Then Resource information is exchanged with the Key Performance Indicator component. For example, Key Performance Indicators for ResponseDelayRate

and ExecutionDelayRate are organized as following figure shows (see Figure 50):

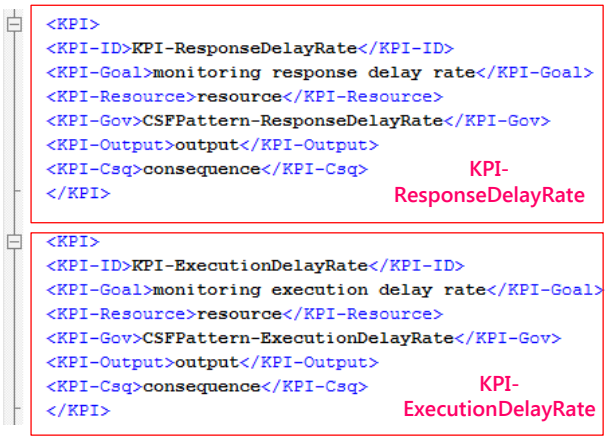


Figure 50 *Excerpt of Use Case KPI*

Each Aggregator is associated to an algorithm for a particular Critical Success Factor and Resource Composition Type. For example, in our Use Case, Aggregators for CSF-ResponseDelayRate are organized depending on different Composition Types defined as: Sequence, Concurrency, Loop and Conditional Branching), as following figure shows (see Figure 51):

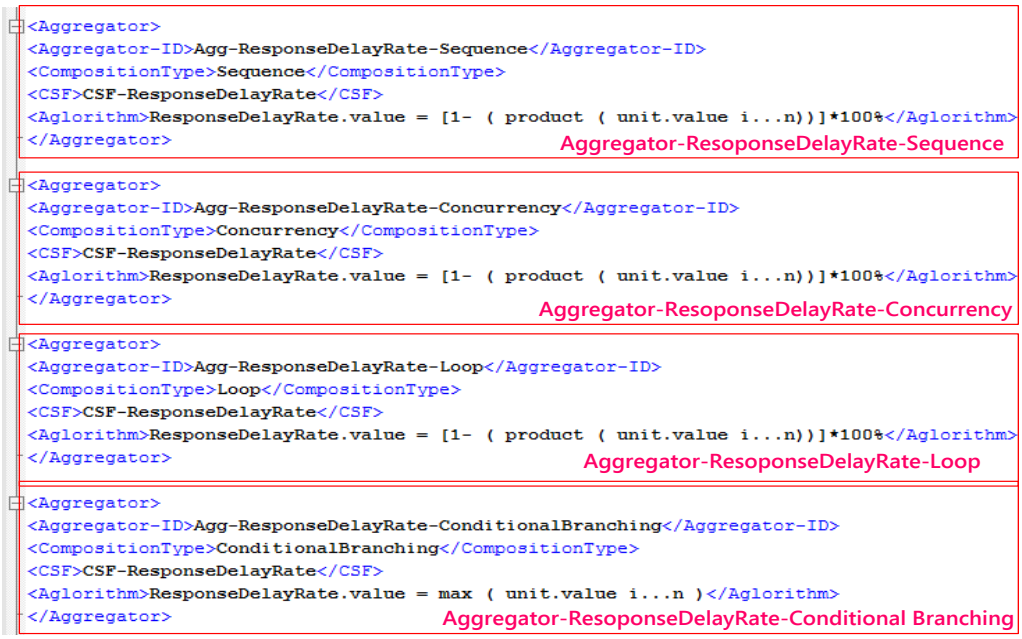


Figure 51 *Excertp of Use Case Aggregator-Algorithm*

4.4 Conclusion

This chapter aimed at answering two research questions:

Q1: what is the object of our governance?

Q2: how the governance objects are organized and what is the interdependence of these objects?

To this end, we introduced our Multi-Layer BaaS Management Framework which includes two key models:

1) Business Resource Organization Model

2) Negotiation and Governance Preparation Model.

The Business Resource Organization Model aims at narrowing the gap between business request and business resources' ability and maximizing the utilization of business resources. This model includes deconstructing and formalizing management requirement and select means to organize convenient resources. The Negotiation and Preparation Model aims at simplifying the negotiation between enterprise and service providers in order to reduce the interruption of agreements' violation, and guarantee the quality of provided BaaS.

In order to achieve this cross-layers governance, we extend the Service Level Agreement (SLA) to Multi-Level Agreements (MLAs) which includes Business Process Level Agreement (BPLA) and Business Service Level Agreement (BSLA). After confirming these Multi-level Agreements, Governance Preparation Process is invoked to organize the essential elements, required by the governance process. We identified six governance elements: (1) Classification of Non Functional Property, 2) Resource Dependency, 3) Transformation Pattern Organization, 4) Governance Pattern Organization, 5) Key Performance Indicator, 6) Aggregator) and present the way they are modelled. Moreover, a logistic Use Case business scenario is built to demonstrate how Business Resource Organization and Governance Preparation Process are implemented.

After defining this governance foundation, we will introduce the proposed Business as a Service Governance Execution and Adapting Framework to resolve our further research questions in the next chapter.

5 Governance Execution and Adapting Framework

5.1 Introduction

In the previous chapter we have introduced our BaaS Management and Governance Preparation Framework. In this chapter will introduce the implementation of the next phases in our Governance Loop: Governance Execution Phase and Adapting Phases thanks to a Governance Execution and Adapting Framework.

This Framework aims at resolving 2 challenges:

- Q3-How can we make the governance approach be customized and self-adjusted to meet the different governance requirements?

- Q4: How governance and adapting processes can be automatic and benefit business outcomes?

Answering these questions requires a customized and dynamic governance process. To implement this governance process, we design an autonomic governance requirement elicitation and governance rule generation process. This elicitation and generation process aims at tuning Business Decision Makers' governance requirements into Platform Independent Policies and Platform Dependent Policies. These generated governance policies and rules are used to orchestrate the business resource Non Functional Property at runtime. Moreover, we take advantage of the functional specification to compose the quality of Non Functional Property and governance policies accordingly. The policy generation process takes advantage of both Model Driven Engineering and of Pattern-based Engineering approach. The transformation strategy relies on a global model that connects business workflow analysis and governance requirements to governance patterns and policy rules. Each resource has its own properties and interfaces with others depending on the business workflow analysis. Non Functional Property requirements are associated to resources. Each requirement is analyzed and transformed into governance rules thanks to a 'Transformation Pattern'. The generated policy rules select, orchestrate and invoke governance execution elements to constrain execution of functional rules by assigning and computing 'Key Performance Indicators'. Governance execution elements can also invoke 'Action Engine (AE)' to tune and improve the resources performance by doing actions on related resources. This allows business activities performance improvement at the runtime. Furthermore, Aggregators can select and aggregate Key Performance Indicators' results

according to the Non Functional Property classification and customized requirements. These aggregated results can be published as mashup reports depending on users' needs. To achieve the governance dynamicity and agility and make governance processes fit the complexity of cloud environment, we adapt immunity inspired autonomic management to control Key Performance Indicators lifecycle evolution. We use our logistic Use Case to illustrate the real-time Monitoring Policy Rule and Computing Rule Generation Process and the implementation of Computing Algorithms.

This chapter is organized as follow: section 5.2 gives a high-level view of our Multi-layer Governance Framework. Then in the section 5.3 we introduce Multi-layer Monitoring Model: Formalization of Governance Monitoring Requirement (section 5.3.1), Generation of Monitoring Policy Rule (section 5.3.2), Implementation of Monitoring Rules (section 5.3.3) and Use Case Structure introduction (section 5.3.4), Use Case – Implementation of Monitoring Policy Rule's Generation (section 5.3.5). Section 5.4 introduces our Multi-layer Computing Model: Formalization of High-level Computing Requirement (section 5.4.1), Generation of Precise Computing Requirement and Computing Rule (section 5.4.2), Implementation of Computing Rule (in section 5.4.3), Key Performance Indicator Evolution and Lifecycle Management (section 5.4.4), Use Case – Implementation of Computing Rule Generation (section 5.4.5) and Use Case – Implementation of Computing Algorithm (section 5.4.6).

5.2 High level view of Multi-layer Governance Framework

As stated section 3.2, there are several steps during the Governance Execution Phase and there are four governance components that are driven by our governance framework. Figure 52 presents the interaction and information exchange between governance components.

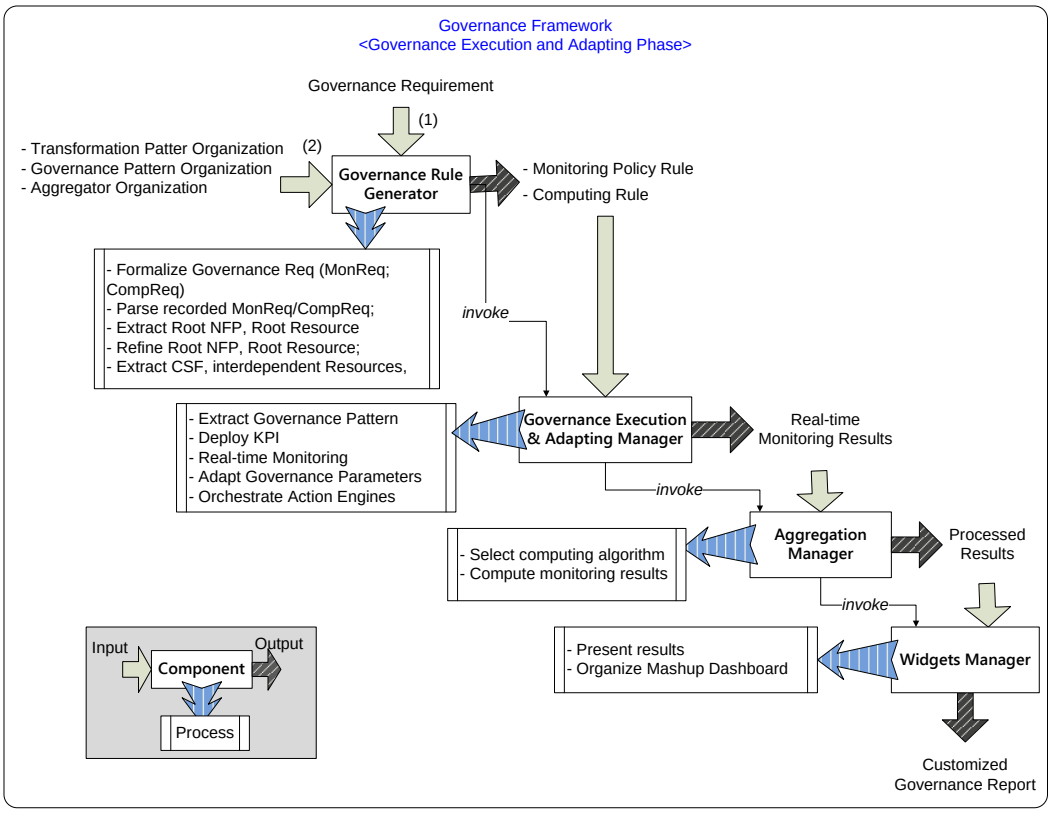


Figure 52 *Interaction of Governance Components*

We can sum up these governance steps into two jobs in the Governance Execution Phase: 1) Real-time Monitoring and 2) Computing & Presentation. This leads us to organize our Multi-layer Governance Framework in two models: 1) Real-time Monitoring Model and 2) Computing Model. In this subsection we introduce the high level view of our governance framework with these two models.

As in the BaaS management framework presented Chapter 4, Business Decision Maker's governance requirements are turned into Governance Rules thanks to Generation Process. These Governance Rules are implemented by orchestrating convenient Governance Elements. This similar approach used for both frameworks allows us to unify data format and keep consistency in our management and governance loop. As presented in chapter 3, we use 2 jobs in our governance framework (namely Monitoring and Computing): Monitoring aims at measuring the performance of governed object at run time, whereas Computing aims at aggregating collected monitoring results to give a comprehensive outcome. Two types of requirement are used:

1) Monitoring Requirements define the Non Functional Property of a particular resource to monitor the quality of particular resource's Non Functional Property;

2) Computing Requirements define the way the collected monitoring results will be computed and composed to sum up the governance results.

Governance requirement (Gov-Req) = Monitoring requirement (MonReq) ∪ Computing requirement (CompReq)) (Equation 37)

Based on these requirements, 2 types of policy rules are generated:

(1) Monitoring rules invoke the convenient Key Performance Indicators to monitor the quality and performance level of a resource's Non Functional Property;

(2) Computing Rules invoke convenient Aggregators to compute comprehensive governance results associated to the collected monitoring information.

Monitoring Policy Rule (PolR) and Computing Rule (CompRule) (Governance Rule = PolR ∪ CompRule). (Equation 38)

The aggregated results provide integrated and comprehensive performance information on business processes. In addition, Action Engines can be invoked to correct the performance level of resource's Non Functional Property and to improve the quality of business resources. Figure 53 shows the high level view of our multi-layer governance model.

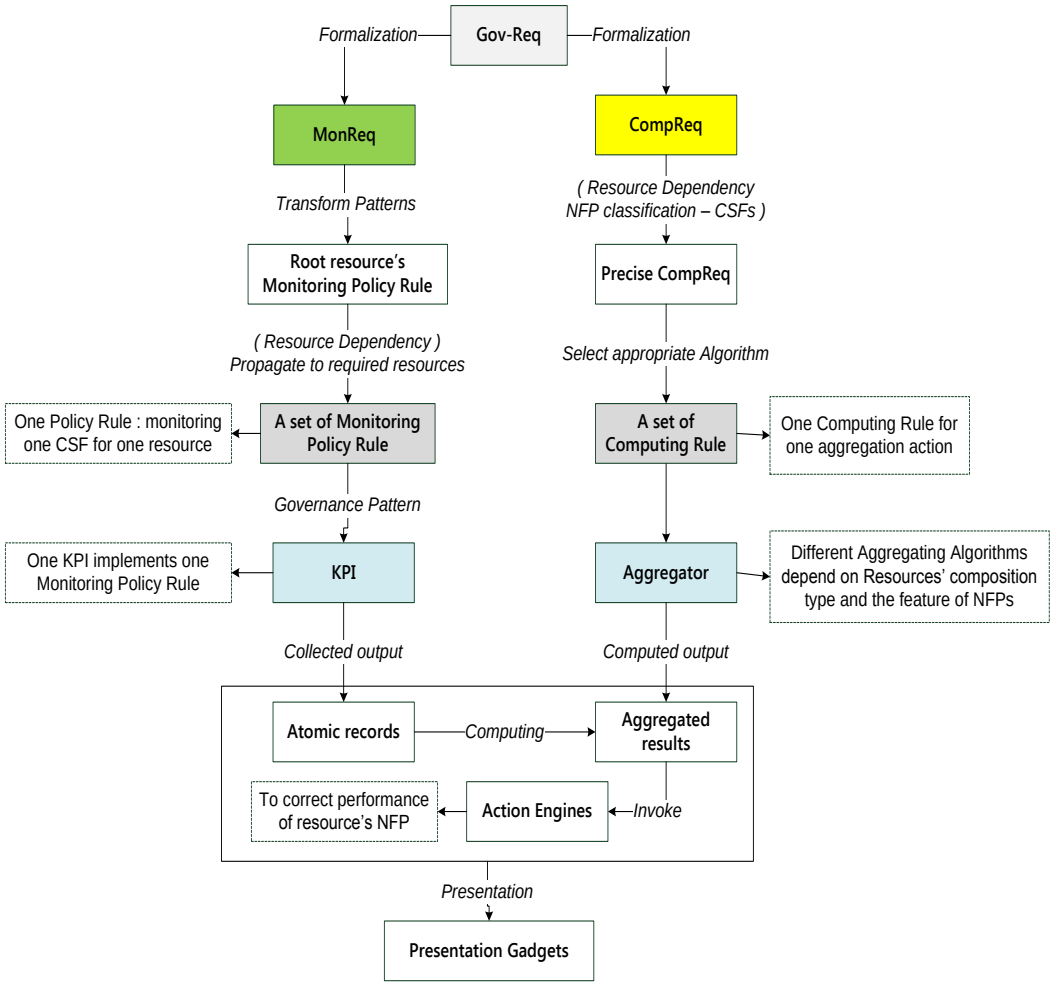


Figure 53 *High level view of Governance Model*

While the monitoring part is managed in a top-down vision, the governance part is achieved using a bottom-up strategy: the composed quality of infrastructure impacts the quality of service, the composed quality of services impacts the quality of related business task, and lastly the composed quality of business task impacts the quality of related business process. Moreover, we consider two types of service: black-box services which focus on monitoring service instance's performance and white box services which take into account the performance of related infrastructure elements and service internal processes. The following Figure 54 shows the overview of this multi-layer governance.

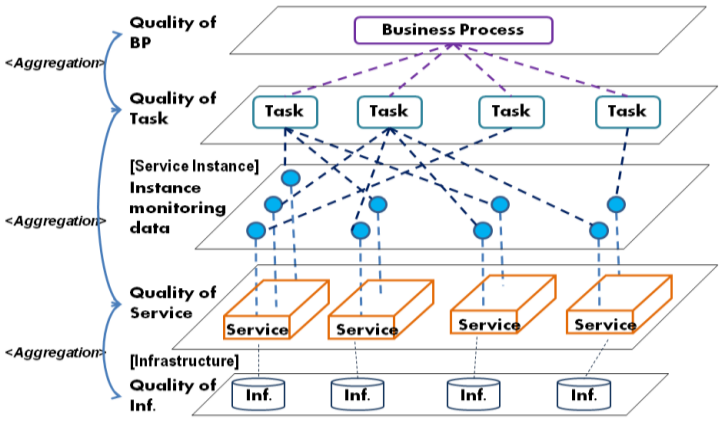


Figure 54 *High Level View of Multi-layer Governance*

After presenting the global organization, we introduce our Multi-layer Monitoring Model section 5.3 and our Multi-layer Computing Model section 5.4.

5.3 Multi-layer Monitoring Model

The essential elements and workflow of the monitoring model are shown Figure 55.

- Formalizing monitoring requirement (MonReq);
- Parsing formalized MonReq to extract Root Resource and Root Non Functional Property;
- Refining Non Functional Property to select convenient specified CSF(Critical Success Factor) Patterns;
- Generating Root Resource's Monitoring Policy Rule (Root PolR);
- Refining Root Resource to extract required low-layer Resources;
- Propagating Root Monitoring Policy Rule to required Resource, and then MonReq's Monitoring Policy Rules (PolRs) are generated.
- Involving generated Monitoring Policy Rules invoke convenient Key Performance Indicators to implement monitoring.

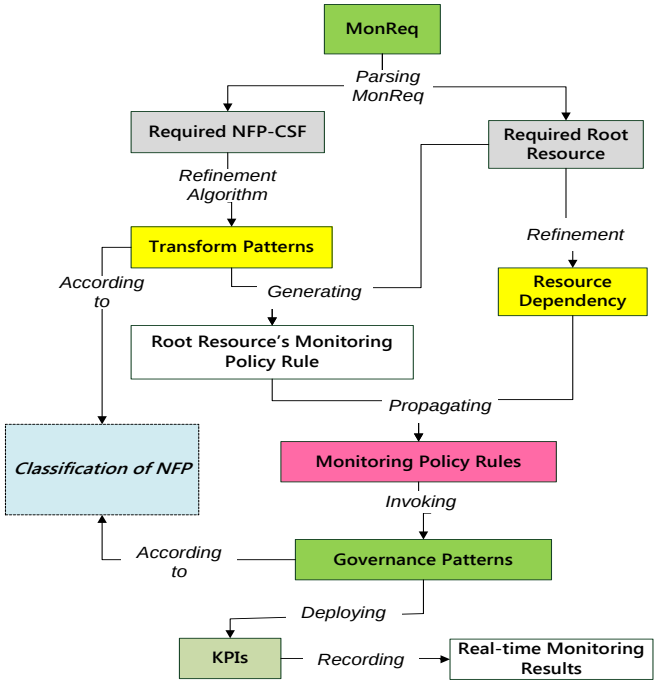


Figure 55 *Multi-layer Monitoring Model*

5.3.1 Formalization of Governance Monitoring Requirement

Our Monitoring Policy Rule generation process is based on a formal model integrating requirements and patterns. By this way policy rules can be designed and transformed in a generic way. As mentioned section 5.2, governance requirement consists of two types of requirements: Monitoring requirements (MonReq) and Computing requirements (CompReq).

Formally, a Monitoring Requirement is defined as a tuple:

MonReq $_{mrq}$ = (MonReq-ID, MonReq-Goal, MonReq-Resource (name/ layer/ type), MonReq-NFP (name/ metrics/ measurement), MonReq-Cxt)

(Equation 39)

Where

- MonReq-ID defines the identity of this monitoring requirement.
- MonReq-Goal defines the objectives of this MonReq;
- MonReq-Resource specifies the target resource's name, layer and type. The target resource can be a Task, a Service, an Operation, or a part of the infrastructure depending on what has been specified in Agreements to be reached in BaaS management process.
- MonReq-NFP (name/metrics/measurement) specifies the target Non Functional Property's name, metrics and measurement which defined in the agreements;

- MonReq-Cxt defines the necessary context for this MonReq.
- 'mrq' is the MonReq's serial number.

We can also get the MonReqs associated to a Resource R_k (MonReq(R_k)) by selecting all the MonReqs which associated Resource (MonReq.R) matches with R_k thanks to the selection function σ:

$$MonReq(R_k) = \sigma(MonReq.R == R_k)(MonReqs). \quad \text{(Equation 40)}$$

5.3.2 Generation of Monitoring Policy Rule

Formalized Monitoring Requirement (MonReq) requires Non Functional Property Refinement according to the Non Functional Property Classification. As we have explained in the Governance Preparation Phase (in section 4.3.4), Transformation Patterns (Pat) are designed to "transform" MonReqs to Root Monitoring Policy Rules by running the Non Functional Property refinement process. After generating Root Monitoring Policy Rule (PolR), the Resource Dependency Refinement Process is launched to extract precise dependent Resources, to propagate Root Monitoring Policy Rule to all related Resources. Lastly this MonReq's Monitoring Policy Rules are generated.

A Policy Rule (either a Root PolR or a Propagated PolR) is also defined as a tuple:

$PolR_x$= (PolR-ID, PolR-G, PolR-R (Resource, Layer), {PolR-Cxt}, PolR-GPat, PolR-S (NFP, Metric, Algorithm)) (Equation 41)

Where

- PolR-ID defines the Policy Rule's identity;
- PolR-G defines the Policy Rule's objectives;
- PolR-R defines which Resource is concerned by this Policy Rule and the layer where this Resource is deployed in.
- PolR-Cxt defines the necessary context of this Policy Rule's implementation;
- PolR-GAPat defines which governance action pattern can be invoked to implement this Policy Rule;
- PolR-S defines the measurement of this Policy Rule. It includes the specification of the Non Functional Property that will be governed, the metrics that are used to measure the Non Functional Property and the algorithm used to implement this Policy Rule.

According to this definition, we can gather all the policy rules of all resources as:

$$PolRs = \cup PolR_x, \quad \text{(Equation 42)}$$

Where 'x' is the Policy Rule's serial number.

Lastly, the set of policy rules attached to any resource R_k (PolRs(R_k)) can be defined by selecting the policy rules which Resource property

(PolRs.PolR-R) matches theresources R_k from the set of the Monitoring Policy Rules (PolRs). This selection process is achieved thanks to the selection function σ.

$$PolRs(R_k) = \sigma(PolRs.PolR\text{-}R==R_k)\ (PolRs). \qquad \text{(Equation 43)}$$

The Schema of a Monitoring Policy Rule's is presented Figure 56.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="ValidPolicyRule">
  <xs:element name="PolicyRule">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="PolicyRule-ID" type="xs:string"/>
        <xs:element name="PolicyRule-Goal" type="xs:string"/>
        <xs:element ref="PolicyRule-Resource"/>
        <xs:element ref="PolicyRule-Governance"/>
        <xs:element name="PolicyRule-GPat" type="xs:string"/>
      </xs:sequence>
      <xs:attribute name="MonReq" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="PolicyRule-Resource">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Resource-ID" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="PolicyRule-Governance">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="NFP-ID" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Property of Monitoring Policy Rule

Object – Resource ID

Object – NFP-CSF

Figure 56 *The Schema of Monitoring Policy Rule*

The generation process starts when users define their requirements using a rather high abstraction level without any implementation technical details. Basic policy rules are generated thanks to a pattern-based transformation process. Our Non Functional Property classification is used to organize transformation pat-terns depending on the Non Functional Property they are related to. Patterns names (PatN) and patterns goals (PatG) are used to identify each pattern.

For each resource, the requirements are turned one after the other in a policy rule. To this end, for a given requirement i associated to a resource R_k, the convenient pattern (Pat) is selected from the pattern set (Pats) thanks to the selection function (σ) that extracts the pattern which name (PatN) matches the requirement associated to the Non Functional Property (MonReq-NFP):

Pat= σ (Pats.PatN == $MonReq_{mrq}$.MonReq-NFP)(Pats); (Equation 44)

Where "mrq" is the serial number of Monitoring Requirement (MonReq).

Then this pattern is used to generate the corresponding policy rule which refers to both the requirement and the resource. Let R_k be the resource associated to the grq^{th} requirement $MonReq_{grq}$, (i.e. R_k = $MonReq_{grq}$.MonReq-NFP-Resource), the policy rule which refers to this requirement and to the kth resource is defined as:

$PolR_{grq-k}$= ($MonReq_{grq}$.MonReq-Resource, Pat.PatN, Pat.PatG, MonReq-Cxt, GAPat); (Equation 45)

After discovering the 'basic policy rule' thanks to this selection process, we have to check the selected pattern's related sub-patterns to get more precise policy rules. If the selected pattern contains at least a related sub-pattern (i.e. when Pat.PatR is not an empty set), a refinement algorithm (see Algorithm 1) is recursively launched to precise and develop the policy rules associated to this pattern. For example, a generic "confidentiality management" pattern can be refined using authentication and authorization patterns as well as encryption sub-patterns.

- Algorithm 1: Refinement algorithm – Transformation Pattern

Objective: select convenient Critical Success Factor patterns
Input: Pattern Pat;
Selection: Ref.PatR(Pat)

If Pat has childpattern;
  Pat = Pat.childpattern;
  Call Ref.PatR(Pat);
*Else {*Pat has no childpattern so Pat is a CSF pattern;}
  Select Pat ;
End

Output: selected CSF pattern=Pat;

At the end of this step the different policy rules associated to the requirements are generated. As for the governance composition, we use a policy composition process, including the functional composition knowledge, to select, extract and compose the different policy rules attached to a resource. Each task identified in the Business Process Layer (BPL) is considered as a sub-Business Process. This involves that they are used to compose / derive the policies associated to the same or lower-layer resources that are used to implement this sub-process. Based on the Resource Dependency Organization which has been implemented in Preparation Phase of our Governance Loop (schema is introduced refer to the Figure 41 in section4.3.3), we use this "Resource dependency" knowledge to select the resources belonging to the lower-layers

involved in the Business Process Layer resource deployment and to propagate these requirements to these Business Service Layer and Business Implementation Layer resources.

As stated in the Business Process Level Agreement (BPLA) and Business Service Level Agreement (BSLA) (introduced in section 4.3.1 and 4.3.2), each requirement is defined by specifying the resource (MonReq-Resource) to which this requirement is associated to the layer to which this resource belongs as well as the goal (MonReq-Goal) and the associated Non Functional Property and metrics (MonReq-NFP).

As a resource R_k ('k' is numbering the resource) can be associated to many requirements, the Computer Independent Model is defined as the set of requirements associated to the different resources:

$$MonReqs = \cup \{MonReqs (R_k)\}. \quad \text{(Equation 46)}$$

After gathering and formatting the requirements in a single Computer Independent Model, the policy generation process consists in turning each CIM assertion in a Platform Independent policy rule. Then the Root Monitoring Policy Rule is propagated to all required dependent Resources. By this way, MonReq's Monitoring Policy Rules are generated.

- Algorithm 2: Refinement algorithm – Resource Dependency
Objective: select required dependent Resource **Input:** Root-Resource Root-res; **Selection:** Dep.Res(Root-res)
If Root-res has required-res; *Then* Root-res = Res.required-res; *Call* Dep.Res(Root-res); *Else* {Root-res has no required-res, Select Res=Root-res}; *Select* Res ; *End*
Output: selected Res=required-res;

5.3.3 Implementation of Monitoring Rules

After generating Monitoring Policy Rules, the Governance Policy Rule Generator will invoke the Governance Action Manager to deploy Key Performance Indicators to implement each Monitoring Policy Rule. Firstly the convenient Governance Pattern will be called to invoke convenient Key Performance Indicators by transporting all necessary information from Monitoring Policy Rules to these Key Performance Indicators. Then selected Key Performance Indicators are invoked to monitor the quality of resource's Non Functional Property and record monitoring results. Governance Patterns

(GPats) are organized according to the classification of Non Functional Properties as we did for Transformation Patterns but Governance Patterns only focus on Critical Success Factors.

The convenient Governance Pattern (GPat) is selected from the governance patterns set (GPats) thanks to the selection function (σ) that extracts the governance pattern which ID (GPat-ID) matches the Policy Rule's property (PolR-GPat):

GPat= σ (GPat.GPat-ID==PolR.PolR-GPat)(GPats); (Equation 47)

After selecting the convenient Governance Patterns, Key Performance Indicators are deployed to monitor resource's Non Functional Properties and to implement generated monitoring policy rules.

A Key Performance Indicator can be assigned for two missions. The basic one is to record and measure implementation of policy rule when the Key Performance Indicator is invoked by governance action patterns. Furthermore, in order to improve the accuracy of governance and allow efficient response to unexpected situations, a Key Performance Indicator is orchestrated as an alert mean. Once an unexpected situation is detected, the Key Performance Indicator will send an alarm signal to the administrator immediately, to make execution system be able to respond quickly.

The convenient Key Performance Indicator is invoked by the selection function σ that matches Key Performance Indicator's ID (KPI_ID) with the Governance Pattern's property (GPat.GPat-KPI):

KPIs= σ (KPIs.KPI_ID==GPat.GPat-KPI)(KPIs). (Equation 48)

We gather all the KPIs' results by appointing a specific resource (R_K)'s Critical Success Factor monitoring result as:

KPI-Outputs(R_K) = σ(KPIs.KPI-Resource==R_K && KPIs.KPI-Gov == 'CSF') (KPIs) (Equation 49)

5.3.4 Use Case Presentation

In this section we introduce our Logistics Company's Business Process Use Case to demonstrate our Governance Process.

There are 3 steps to development BaaS governance processes as Figure 57.

- Step 1: Generate Monitoring Policy Rules
- Step 2: Implement Monitoring Policy Rules
- Step 3: Generate Computing Rules and implement aggregation.

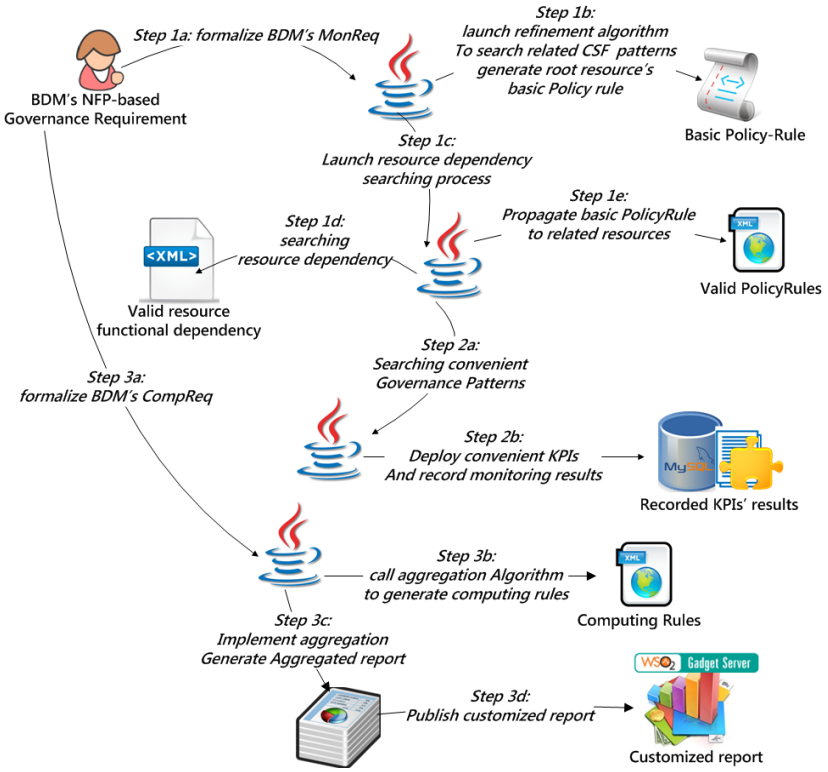


***Figure 57** Organization of BaaS Management Case Study*

After building the customized business scenario (in section 4.2.5) associated to a Distribution Business Process and refining governance elements' organization in Governance Preparation Phase (in section 4.3.6 – 4.3.8), we present here the Monitoring Policy Rule's Generation Process.

The governance requirement: "in a given time period (one week), the Business Process' delay rate." is formalized as:

- *Monitoring requirement 1= ("MonReq 1", "monitoring Express Delivery BP's delay rate", "Express Delivery BP", "Delay rate", "Cxt=express");*

- *Monitoring requirement 2 = ("MonReq 2", "monitoring Standard Delivery BP's delay rate", "Standard Delivery BP", "Delay rate", "Cxt=standard");*

These two Monitoring Requirements (MonReqs) are recorded in < MonReq.xml >. To generate Monitoring Policy Rules for these two requirements, this xml file is parsed to extract Root Resource and Root Non Functional Property (see Figure 58).

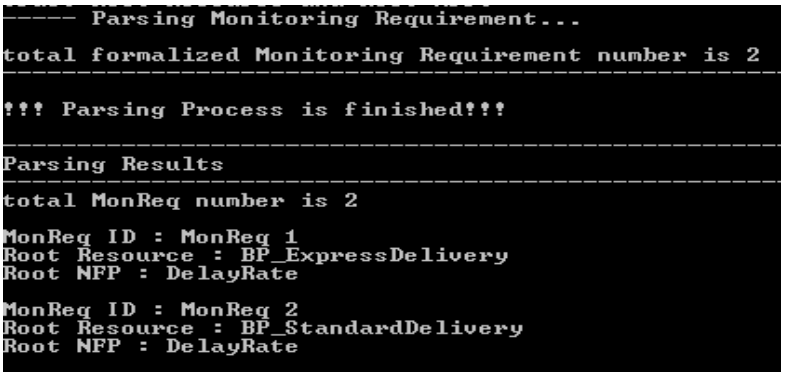


Figure 58 *Use Case - Parsing Monitoring Requirements*

For each Monitoring Requirement the Non Functional Property refinement process is launched to extract convenient Critical Success Factors for the Root Non Functional Property. As a result, there are two Critical Success Factors Patterns are refined: Response Delay Rate and Execution Delay Rate (see Figure 59):

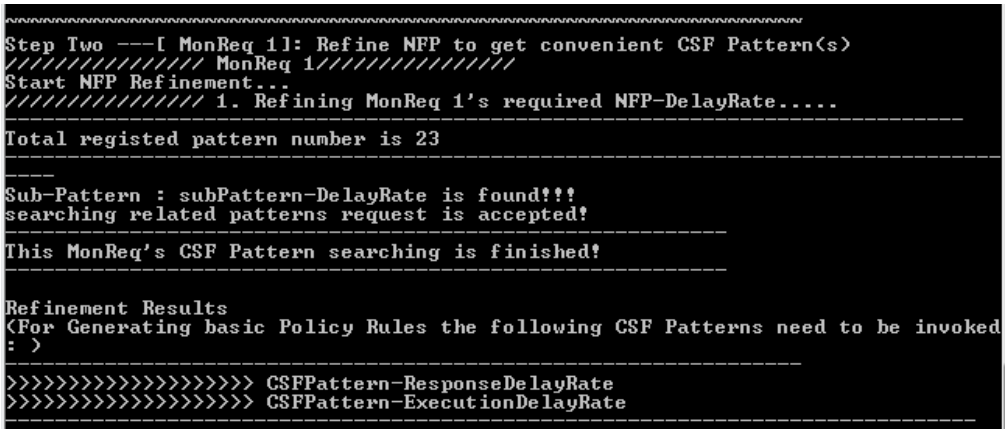


Figure 59 *Use Case - NFP Refinement Process*

Due to the extracted Critical Success Factors, the convenient Governance Patterns are selected. Root Resource's Monitoring Policy Rule (Root-PolR) is generated in < Root-MonitoringPolicyRule.xml >. Two Root PolRs are generated from MonReq 1 (see Figure 60 and Figure 61).

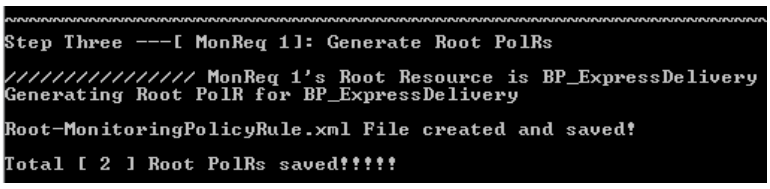


Figure 60 *Use Case -Root Monitoring Policy Rule's Generation*

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<ValidPolicyRule>
  <PolicyRule MonReq="MonReq 1">
    <PolicyRule-ID>PolR-BP_ExpressDelivery-1383841397996</PolicyRule-ID>
    <PolicyRule-Goal>Monitoring BP_ExpressDelivery's CSFPattern-ResponseDelayRate</PolicyRule-Goal>
    <PolicyRule-Resource>
      <Resource-ID>BP_ExpressDelivery</Resource-ID>
    </PolicyRule-Resource>
    <PolicyRule-Governance>
      <NFP-ID>ResponseDelayRate</NFP-ID>
    </PolicyRule-Governance>
    <PolicyRule-GPat>GPat-ResponseDelayRate</PolicyRule-GPat>
  </PolicyRule>
  <PolicyRule MonReq="MonReq 1">
    <PolicyRule-ID>PolR-BP_ExpressDelivery-1383841398020</PolicyRule-ID>
    <PolicyRule-Goal>To monitor BP_ExpressDelivery's CSFPattern-ExecutionDelayRate</PolicyRule-Goal>
    <PolicyRule-Resource>
      <Resource-ID>BP_ExpressDelivery</Resource-ID>
    </PolicyRule-Resource>
    <PolicyRule-Governance>
      <NFP-ID>ExecutionDelayRate</NFP-ID>
    </PolicyRule-Governance>
    <PolicyRule-GPat>GPat-ExecutionDelayRate</PolicyRule-GPat>
  </PolicyRule>
</ValidPolicyRule>
```

Figure 61 *Use Case - Generated Root-PolR*

According to the definition, the generated Root Monitoring Policy Rule (Root-PolR) from the monitoring requirement MonReq1 can be organized as:

- Root-PolR 1 = ("PolR 1","count response delay time", "Resource(Express Delivery BP, BPL, white-box)", "", "Gov(ResponseDelayRate, count delay time out of measured time)", "GPat-RespDelay");

- Root-PolR 2 = ("PolR 2", "count execution delay time", "Resource(Express Delivery BP, BPL, white-box)", "", "Gov(ExecutionDelayRate, count delay time out of measured time)", "GPat-ExecDelay").

More details for the generated Root-PolRs are presented in Appendix chapter 3.

After generating Root-PolR, the next step is the Resource Dependency Refinement Process is launched to extract dependent resources for Root Resource. The following figure shows the refinement results (Figure 62).

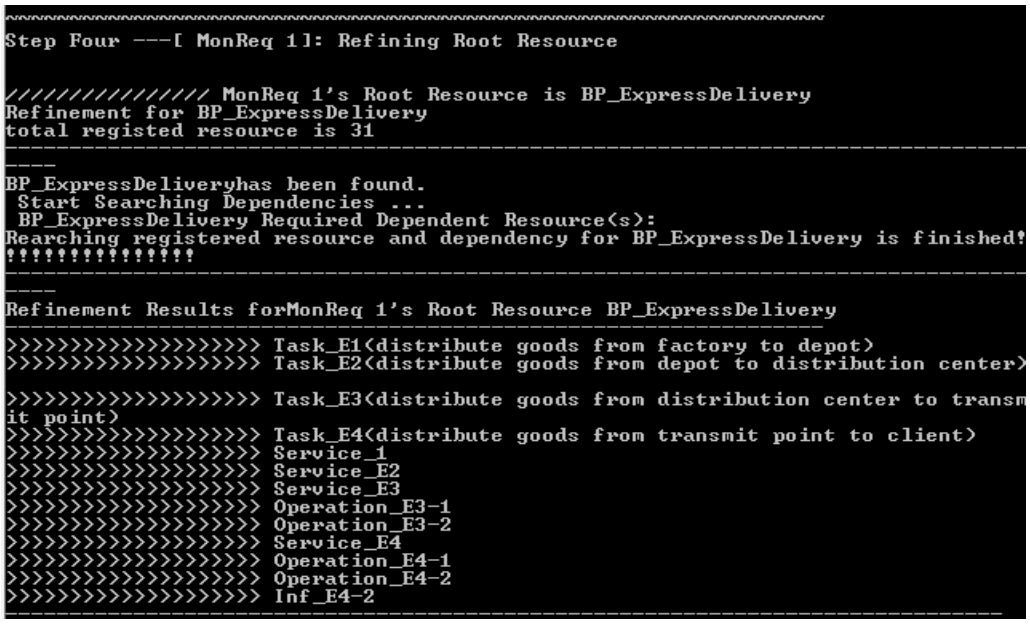


Figure 62 *Use Case - Resource Refinement*

The final step for generating MonReq1's Monitoring Policy Rule is to propagate the Generated Root-PolR to Resources found after the refinement process. 26 propagated rules are generated in <Propagate-MonitoringPolicyRule.xml > file (See Figure 63).

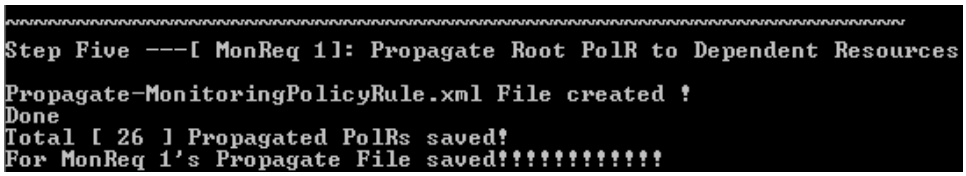


Figure 63 *Use Case - Propagate PolR results*

These generated Monitoring Policy Rules are given in Appendix chapter 3. For example, the Root Monitoring Policy Rule (Root-PolR 1) is propagated to the associated resource Task 4, Service E4, Operation E4-1, Operation E4-2 and Infrastructure element E4-2. Then these rules are defined as:

1) Root-PolR 1 is propagated to Task 4:

- *PolR 1_6 = ("PolR 1_6", "count response delay time", "Resource(Task_E4, BPL, white-box)", "", "Gov(response delay time, count delay time out of measured time)", "GPat-RespDelay")*

1-a) Root-PolR 1 is propagated to Service_E4:

- PolR 1_10 = ("PolR 1_10", "count response delay time", "Resource(Service_E4, BSL, white-box)", "", "Gov(response delay time, count delay time out of measured time)", "GPat-RespDelay")

1-b) Root-PolR 1 is propagated to Operation_E4-1 and Operation_E4-2

- PolR 1_13 = ("PolR 1_13", "count response delay time", "Resource(OP_E4-1, BSL, black-box)", "", "Gov(response delay time, count delay time out of measured time)", "GPat-RespDelay")

- PolR 1_14 = ("PolR 1_14", "count response delay time", "Resource(OP_E4-2, BSL, white-box)", "", "Gov(response delay time, count delay time out of measured time)", "GPat-RespDelay")

1-c) Root-PolR 1 is propagated to Infrastructure_E4-2:

- PolR 1_15 = ("PolR 1_15", "count response delay time", "Resource(Inf_E4-2, BSL, black-box)", "", "Gov(response delay time, count delay time out of measured time)", "GPat-RespDelay")

After generating the policy rule set, the Governance Pattern (GPat) and Key Performance Indicator organization file are parsed to invoke convenient Governance Patterns and Key Performance Indicator s. Following figures show the parsed information from registered Governance Patterns and Key Performance Indicators in organization files (Key Performance Indicator Repository.xml and GovernancePattern.xml). In our Use Case there are total 14 registered Governance Patterns and Key Performance Indicator s according to the organization of Non Functional Property's Critical Success Factors (See Figure 64).

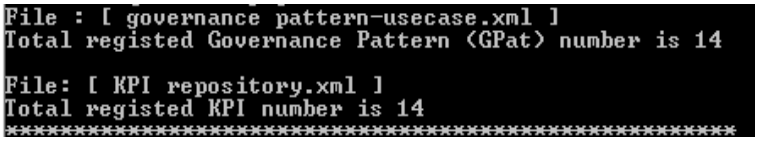


Figure 64 *Use Case - GPat and KPI info*

The generated Monitoring Policy Rules focus on Response Delay Rate and Execution Delay Rate (see Figure 65 showing the definition of related Governance Patterns and Key Performance Indicators).

```
<GovernancePattern>
  <GPat-ID>GPat-ResponseDelayRate</GPat-ID>
  <GPat-Goal>monitoring response delay rate</GPat-Goal>
  <GPat-Context/>
  <GPat-KPI>KPI-ResponseDelayRate</GPat-KPI>
  <GPat-Trans>
    <PolR-resource>resource</PolR-resource>
    <PolR-Gov>CSFPattern-ResponseDelayRate</PolR-Gov>
  </GPat-Trans>
  <GPat-Csq/>
</GovernancePattern>
<GovernancePattern>
  <GPat-ID>GPat-ExecutionDelayRate</GPat-ID>
  <GPat-Goal>monitoring execution delay rate</GPat-Goal>
  <GPat-Context/>
  <GPat-KPI>KPI-ExecutionDelayRate</GPat-KPI>
  <GPat-Trans>
    <PolR-resource>resource</PolR-resource>
    <PolR-Gov>CSFPattern-ExecutionDelayRate</PolR-Gov>
  </GPat-Trans>
  <GPat-Csq/>
</GovernancePattern>
```

```
<KPI>
<KPI-ID>KPI-ResponseDelayRate</KPI-ID>
<KPI-Goal>monitoring response delay rate</KPI-Goal>
<KPI-Resource>resource</KPI-Resource>
<KPI-Gov>CSFPattern-ResponseDelayRate</KPI-Gov>
<KPI-Output>output</KPI-Output>
<KPI-Csq>consequence</KPI-Csq>
</KPI>

<KPI>
<KPI-ID>KPI-ExecutionDelayRate</KPI-ID>
<KPI-Goal>monitoring execution delay rate</KPI-Goal>
<KPI-Resource>resource</KPI-Resource>
<KPI-Gov>CSFPattern-ExecutionDelayRate</KPI-Gov>
<KPI-Output>output</KPI-Output>
<KPI-Csq>consequence</KPI-Csq>
</KPI>
```

Figure 65 *Use Case – Convenient Governance Pattern and KPI*

After generating the Monitoring Policy Rules for the first requirement, a same generation process is launched for the second requirement (more details in Appendix chapter 3), until all registered MonReqs' PolRs are generated. To track this generation process a < PolR-Activation-Log.xml > file is generated to record all generation information (see Figure 66).

Figure 66 *Use Case - PolR-Activation-Log*

According to the resource organization, the Express Delivery Business Process has been decomposed into 7 monitored nodes (see Figure 67): Service 1, Service E2, Operation E3-1, Operation E3-2, Operation E4-1, Operation E4-2, Infrastructure element E4-2. Monitoring Policy Rules have been generated and deployed for each monitored node.

More details for the Standard Delivery Business Process' monitored nodes are given in Appendix chapter 3.

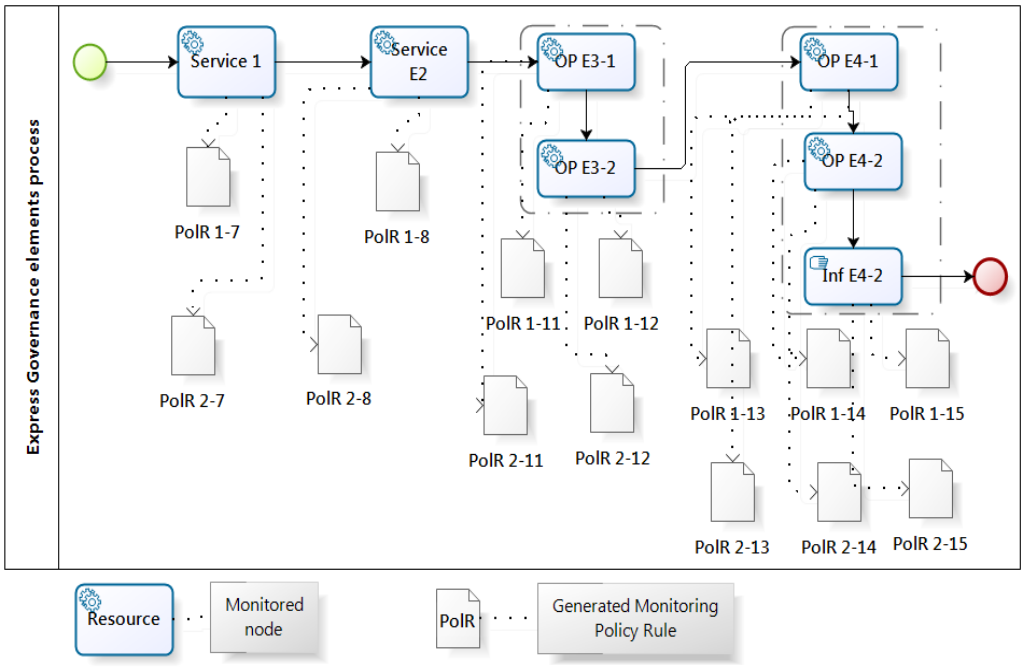


Figure 67 *Express Delivery BP's Monitoring Element (Process/Monitoring Policy Rules/ KPI)*

To implement these monitoring policy rules, convenient Governance Patterns are required to deploy Key Performance Indicators on these monitoring policy rules. Governance Patterns (GPats) are organized according to Non Functional Properties' Critical Success Factors organization, Governance Patterns aim at orchestrating and assigning convenient Key Performance Indicators to implement monitoring policy rules.

Each monitoring policy rule has to invoke a Key Performance Indicator to monitor its implementation. Following Key Performance Indicators are assigned to implement Monitoring Policy Rules for Express Delivery Business Process' required monitoring elements (see Figure 68):

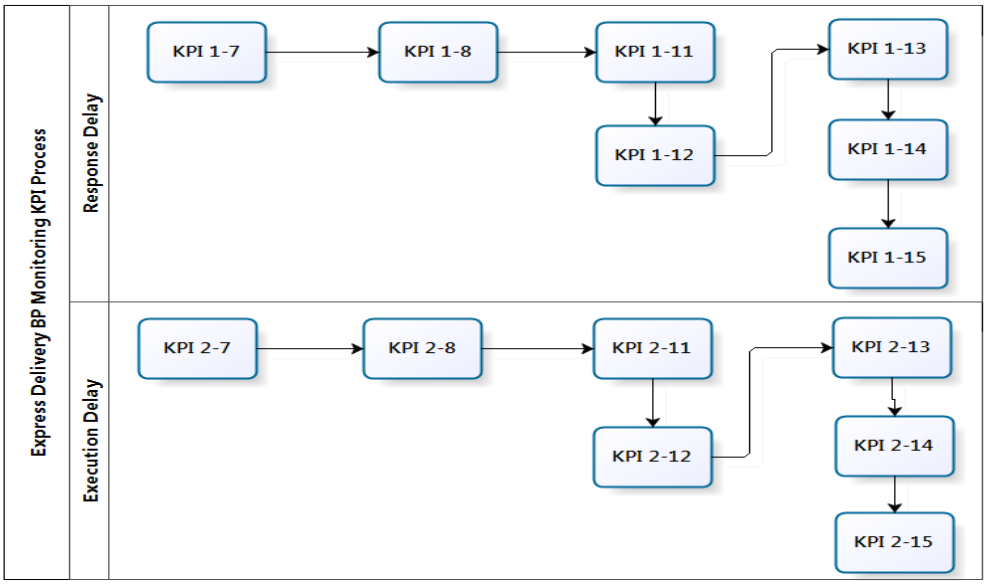


Figure 68 *Organization of KPIs for Express Delivery BP*

We list the definition of Key Performance Indicators for "Task 4". More details for the definition of Key Performance Indicators are given in Appendix chapter 3.

a) Key Performance Indicators for Response Delay Rate for Task4' related Resources:

- Key Performance Indicator 1-13 = ("KPI 1-13", "monitor response delay", "resource (OP E4-1/black-box/BSL)", "CSF (Response delay rate, count delay time out of measured time)", "Output(distance, data, Timestamp)", "consequence{}");

- KPI 1-14 = ("KPI 1-14", "monitor response delay", "resource (OP E4-2/black-box/BSL)", "CSF (Response delay rate, count delay time out of measured time)", "Output(distance, data, Timestamp)", "consequence{}");

- KPI 1-15 = ("KPI 1-15", "monitor response delay", "resource (Inf E4-2/black-box/BIL)", "CSF (Response delay rate, count delay time out of measured time)", "Output(distance, data, Timestamp)", "consequence{}");

b) KPIs for Execution Delay Rate for Task4's related Resources:

- KPI 2-13 = ("KPI 2-13 ", "monitor execution delay", "resource (OP E4-1/black-box/BSL)", "CSF (Execution delay rate, count delay time out of measured time)", "Output(distance, data, Timestamp)", "consequence{}");

- KPI 2-14 = ("KPI 2-14 ", "monitor execution delay", "resource (OP E4-2/black-box/BSL)", "CSF (Execution delay rate, count delay time out of measured time)", "Output(distance, data, Timestamp)", "consequence{}");

- KPI 2-15 = ("KPI 2-15", "monitor execution delay", "resource (Inf E4-2/black-box/BIL)", "CSF (Execution delay rate, count delay time out of measured time)", "Output(distance, data, Timestamp)", "consequence{}");

Then to fulfill Business Decision Maker's Monitoring requirements these defined Key Performance Indicators are deployed.

5.4 Multi-layer Computing Model

Our computing model aims at satisfying Business Decision Maker's computing requirements to compute related Key Performance Indicator results and then present comprehensive governance reports. In this section we introduce elements and working process of this computing model (see Figure 69).

- Parsing high-level Computing Requirement (CompReq) extracts Root Resource and Root Non Functional Property;
- Refining Root Resource selects required Resources;
- Refining Root Non Functional Property selects convenient Critical Success Factor;
- Generate Precise Computing Requirement (Precise-CompReq);
- Select convenient Computing Aggregator according to the Non Functional Property feature and the Resource composition type;
- Generate Computing Rules (CompRules);
- Implement Computing Rules to present computed results in customized mashup dashboard.

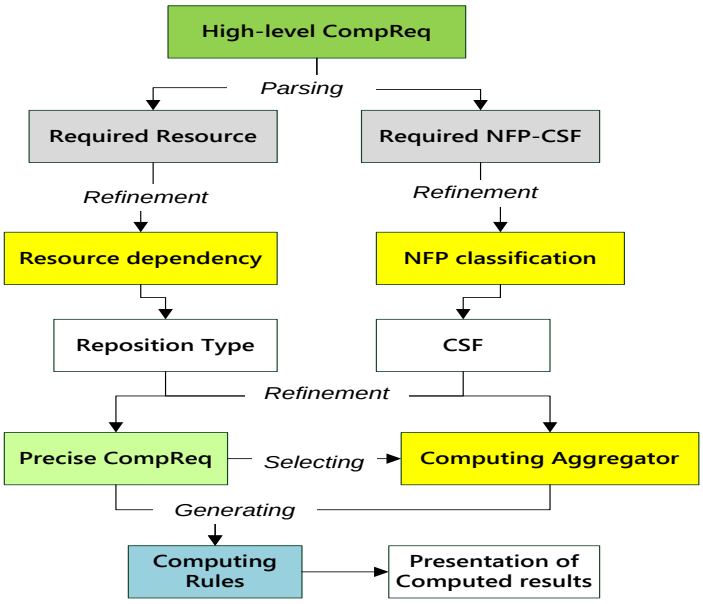


Figure 69 *The Main Elements of the Computing Model*

5.4.1 Formalization of High-level Computing Requirement

Business Decision Maker's governance requirement (Gov-Req) is divided into two types: Monitoring Requirement (MonReq) and Computing Requirement (CompReq):

GovReqs = MonReqs ∪ CompReqs. (Equation 50)

As we did with Monitoring Requirement (in section 5.3.1), a formalized Computing Requirement is also defined as a tuple:

CompReq = (CompReq-ID, CompReq-Goal, CompReq-Resource (name/type/layer), CompReq-CSF (name/metrics/Measurement), CompReq-CompositionType) (Equation 51)

Where

- CompReq-ID defines computing requirement's identity;
- CompReq-Goal defines the objectives of the computing requirement;
- CompReq-Resource (name/type/layer) defines the target resources' essential information;
- CompReq-CSF (name/metrics/Measurement) defines the target Critical Success Factor's information;
- CompReq-CompositionType defines the resources' composition type. It allows aggregation process to call the convenient aggregation algorithm.

We use the traditional four composition types (viz. Sequence, Concurrency, Conditional Branching, Loop) to describe all defined Resources' composition type (see Figure 70 for the notation).

- Sequence: All Resources' actions occurred in series. The Resource former action invokes latter Resource's action. The former result is needed by later Resource. In other words, the failure of any service results in the failure of the composite workflow.
- Concurrency: Two or more Resources' actions occurred in parallel, they are invoked concurrently. Usually these Resources are functionally complementary. In this case there is a need for synchronization.
- Conditional Branching: Only one Resource out of several Resources is invoked depending on the result of branching conditions.
- Loop: One or more Resources' actions can be part of a "cycle" in the composite workflow. We can take this loop composition as a special case of Sequence. These Resources will be invoked as many times as needed until the given loop condition is satisfied.

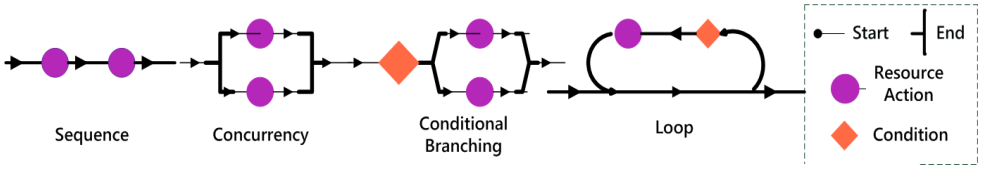


Figure 70 *Four Sorts of Resource Composition*

According to the different features of these four Resource Composition types and to the Critical Success Factor feature, different Aggregating Algorithms are required. Precise Aggregation Algorithms will be introduced in section 5.4.3.

5.4.2 Generation of Precise Computing Requirement and Computing Rule

Before generating Computing Rule, the formalized high-level Computing Requirements (CompReq) need to be refined to define precise Computing Requirement. The refinement process includes Resource Refinement and Non Functional Property refinement. Root Resource and Root Non Functional Property can be extracted by parsing formalized high-level computing requirement CompReq. Root Resource's refinement has been introduced in section 5.3.2 (Algorithm 2). Non Functional Property refinement algorithm (see Algorithm 3) is similar to the Transformation Pattern Refinement algorithm (see section 5.3.2 Algorithm 1).

- Algorithm 3: Refinement algorithm – Non Functional Property

Objective: select convenient Critical Success Factor
Input: Root-NFP nfp;
Selection: Ref.CSF(nfp)

If Root-NFP has childNFP;
Then
  nfp = nfp.childNFP;
  Call Ref.CSF(nfp);
Else nfp has no childNFP so nfp is a CSF;
  Select CSF ;
End

Output: selected CSF = nfp;

After running the refinement process and confirming Resource Composition Type, precise Computing Requirements are generated (see Figure 71).

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="PreciseComputingRequirements">
  <xs:element name="PreciseCompReq">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="PreciseCompReq-ID" type="xs:string"/>
        <xs:element name="PreciseCompReq-Goal" type="xs:string"/>
        <xs:element ref="PreciseCompReq-Resources"/>
        <xs:element name="PreciseCompReq-CompositionType" type="xs:string"/>
        <xs:element name="PreciseCompReq-CSF" type="xs:string"/>
      </xs:sequence>
      <xs:attribute name="CompReq" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="PreciseCompReq-Resources">
</xs:schema>
```

Figure 71 *Schema of Precise Computing Requirement*

Composition algorithms are designed according to Resource Composition Type and Critical Success Factor feature. After generating Precise Computing Requirement (PreciseCompReq), Aggregator-Algorithms can be selected by matching computing requirement's PreciseCompReq-CompositionType with Aggregator's CompositionType and matching computing requirement's PreciseCompReq-CSF with Aggregator's Critical Success Factor.

Aggregator-Algorithm = σ (PreciseCompReq.CompReq-CompositionType == Aggregator.CompositionType && PreciseCompReq.CSF == Aggregator.CSF) Aggregators (Equation 52)

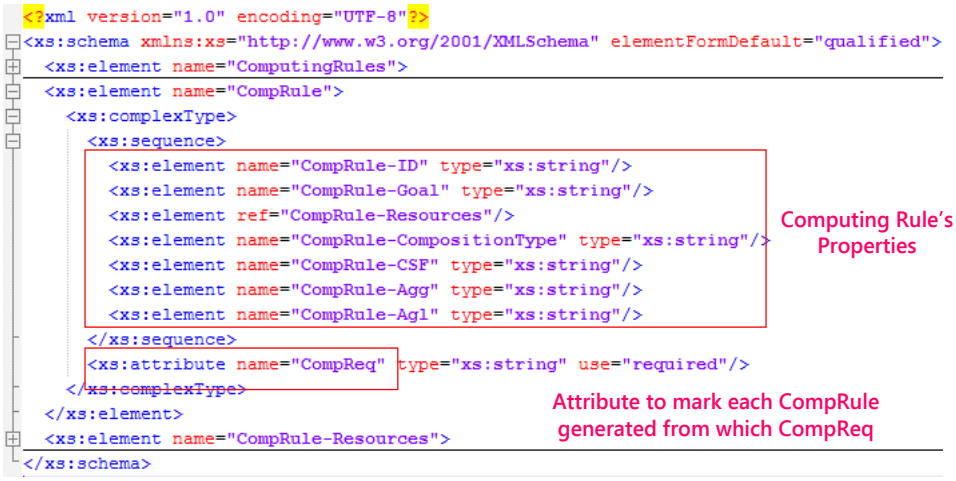
After selecting the appropriate Aggregator, Computing Rules are generated. A Computing Rule is defined as a tuple in a similar way as Monitoring Policy Rule:

CompRule = (CompRule-ID, CompRule-Data (resource, CSF, value, timestamp), CompRule-CompositionType, CompRule-Algorithm, Aggregator) (Equation 53)

Where

- CompRule-ID defines computing rule's identity;
- CompRule-Data (resource, CSF, value, timestamp) defines all computing required data information;
- CompRule-CompositionType defines this computing rule's related resource's composition type;
- CompRule-Algorithm defines required computing algorithm;
- Aggregator defines this computing rule requires aggregator;

Figure 72 shows the Computing Rule's definition schema.




```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="ComputingRules">
  <xs:element name="CompRule">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="CompRule-ID" type="xs:string"/>
        <xs:element name="CompRule-Goal" type="xs:string"/>
        <xs:element ref="CompRule-Resources"/>
        <xs:element name="CompRule-CompositionType" type="xs:string"/>
        <xs:element name="CompRule-CSF" type="xs:string"/>
        <xs:element name="CompRule-Agg" type="xs:string"/>
        <xs:element name="CompRule-Agl" type="xs:string"/>
      </xs:sequence>
      <xs:attribute name="CompReq" type="xs:string" use="required"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="CompRule-Resources">
</xs:schema>
```

Figure 72 *Schema of Computing Rule*

5.4.3 Implementation of Computing Rule

As mentioned section 5.4.1, Aggregation Algorithms depend on Resource Composition Type and Critical Success Factor feature. We defined a generic procedure for aggregating quality of Non Functional Property from selected Key Performance Indicators' results (see Algorithm 4).

- **Algorithm 4 : Aggregation Procedure**

Objective: Aggregate selected Key Performance Indicator Results accordingly

Inputs: A set of selected KPI-Output;

Process: Aggregation Algorithm;

// choosing the appropriate algorithm depends on Resource Composition type and feature of Non Functional Property

Outputs: Aggregated Result

Begin

Identify Non Functional Property;
Call convenient Aggregation Algorithm;
Aggregate Quality of Non Functional Property;
AggProcedure (KPI-Output, ActionCompositionType)
{
    If (ActionCompostionType == sequence)
    {   SeqAgg(KPI-Output)
        agg.value= Σ KPI-Output;}
    If (ActionCompositionType == concurrency)

```
{    ConAgg(KPI-Output)
   // (depends on specific Non Functional Property):
      agg.value=Σ KPI-Output;
  or
      agg.value= max (atomoc.value1…atomic.value n);}
If (ActionCompostionType == condition branching)
{     CBAgg(KPI-Output)
     agg.value= max(atomoc.value1…atomic.value n);}
If (ActionCompositionType == loop)
{     LoopAgg(KPI-Output)
     agg.value= n* atomic.value;}
}
End
```

To aggregate Key Performance Indicators' results to get a meaningful comprehensive governance result, we need to consider the specific Aggregation Algorithms that depend on the different Resource Composition Types.

Our Non Functional Property's aggregation catalogue (see Figure 73) uses the Non Functional Property taxonomy introduced section 3.3.

Base on this generic aggregation process, we define different aggregation algorithms for the most required Non Functional Properties (viz. Price, Response Time, Reputation, Delay Time, Availability, Reliability, Usability, Accuracy, Security and Failure Rate). According to this definition, the convenient aggregator can be invoked by using the selection function σ to select the Computing Rule's Aggregator that matches the Aggregator's Agg-ID:

Aggregator (Agg-ID) = σ (CompRule.Aggregator == Aggregator.Agg-ID) Aggregators. (Equation 54)

An example of Non Functional Property (Delay Rate) aggregation algorithm is presented in following. More details of Non Functional Properties' aggregation algorithms are given in Appendix chapter 4. The specifications include a definition, a data structure and an algorithm.

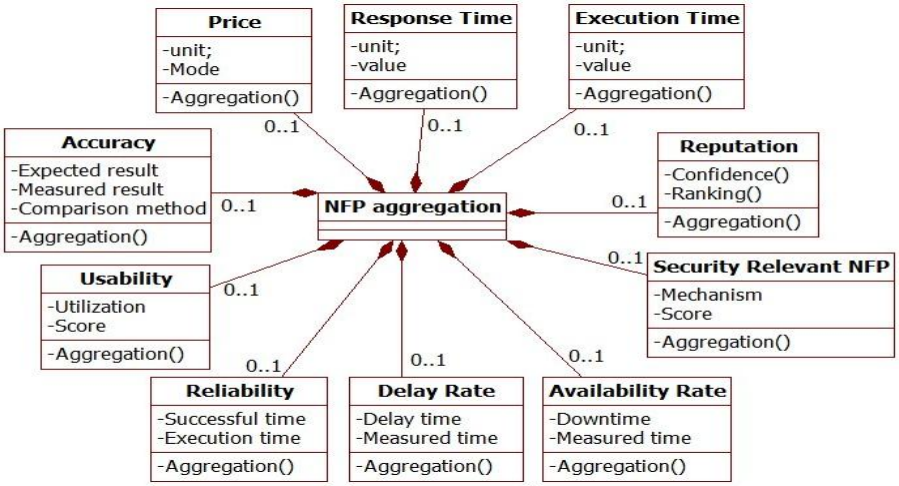


Figure 73 *NFP Aggregation Classification*

Example of an Aggregation Algorithm-Delay Rate:

Description: It is the percentage of delayed service's quantity out of total invoked service's quantity in a given time period.

Data Structure: it includes five elements: time unit, a given time period, amount of delayed time, amount of measured time and value.

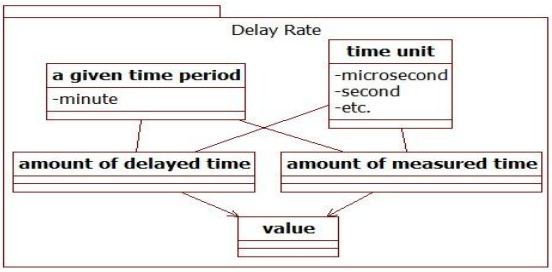


Figure 74 *Data Structure of Delay Rate*

The Algorithm Delay Rate in a given time period is defined as:

DelayRate = (delayed time/ measured time)*100% (Equation 55)

Algorithm 1: [1- ($\prod_{i=1}^{n} DelayRate\ i$)]*100% (Equation 56) is used for Sequence, Concurrency and Loop composition types;

Algorithm 2: Max ($DelayRate_i$ … $DelayRate_n$) (Equation 57) is used for Conditional Branching type. In order to guarantee the quality of BaaS, we take the maximum delay rate of involved Resources as the Conditional Branching type's delay rate.

5.4.4 Key Performance Indicator Evolution and Lifecycle Management

As section 3.3 discussed, our governed objects include Business Processes, Tasks, Services, Operations and Infrastructure elements. Governing a high level Business Process requires a composition of associated lower levels governance. Due to the complexity of collaborative business context, Business Decision Maker should manage a large size of governed object. Moreover, our governance solution focuses on the Non Functional Properties of each governed object. According to our Non functional property classification, each Non Functional Property group includes sub-Group Non Functional Property and a set of Critical Success Factor. To specify governance, we generate refined Monitoring Policy Rule for each governed object's each Critical Success Factor. Each Monitoring Policy Rule invokes associated a Key Performance Indicator to monitor performance level of the governed object. As a consequence, high level governance requirements can be refined to generate lots of Monitoring Policy Rules, and then lots of Key Performance Indicators need to be managed. Furthermore, due to the changing of business requirements and quality ranges, it requires to update Key Performance Indicator adjusting quality range and eliminating useless Key Performance Indicators.

To simplify Key Performance Indicator management and to allow Business Decision Makers to make decision efficiently without getting interference from massive useless information, we take advantage of genetic algorithm to design a Key Performance Indicator Evolution process. This process aims at eliminating useless Key Performance Indicators and reducing monitoring errors to allow governance management to focus on the most active Key Performance Indicators and getting accurate monitoring results. Respecting the biological immunity process, we divide a Key Performance Indicator's monitoring results into two types: self and non-self.

- A "self" result means it is a satisfied result according to governance agreement. A "distance" is defined to measure the difference between a monitoring result and the associated required result. If a "distance=D[0] means this monitoring result matches the associated required result, this is a satisfied result (self).

The "self" is defined as "self = KPI_result [distance=D[0]]".

(Equation 58)

- A "non-self" result means it is an unsatisfied result which should trigger an alert to arise the attention of Business Decision Makers. The "distance" between "non-self" and the associated required result is defined as "distance=D[1]".

The "non-self" is defined as "non-self = KPI_result [distance=D[1]]".

(Equation 59)

All monitoring performance results (AP) are composed by "self" and "non-self":

Self ∪ Nonself = AP; Self ∩ Nonself = $\boldsymbol{\phi}$; (Equation 60)

Due to the complexity of collaborative business context, too many False Positive Errors and False Negative Errors can make the monitoring meaningless. We define a False Positive Error is that a Key Performance Indicator alarms a satisfied result as an unsatisfied result. A False Negative Error is a Key Performance Indicator detects an unsatisfied result as a satisfied result. These errors usually occur when Business Decision Makers change their business requirements and re-define quality ranges. In a dynamic business context, it requires to avoid these monitoring errors. To address this problem and to achieve flexible and autonomic Key Performance Indicators management, we define the Key Performance Indicator's evolution process includes three strategies:

1) Define a four-stage Key Performance Indicator lifecycle,
2) Define a self-tolerance strategy,
3) Define a self-variation strategy.

The four-stage lifecycle aims at managing Key Performance Indicator Evoluation automatically and eliminating those which make enough False Negative Errors. These lifecycle stages include (Figure 75):

A) Immature KPI (I_{KPI}): initialized Key Performance Indicator which is invoked after required business scenario is built. If a I_{KPI} passed "self-tolerance", it can evolve to the Mature stage. Otherwise, if it alarms "self", it goes to Dead stage.

B) Mature KPI (T_{KPI}): Key Performance Indicator has passed self-tolerance which has a fixed mature lifetime. A Mature Key Performance Indicator detects and alarms enough "non-self" it can evolve to Memory stage. Otherwise, if it is too old or it alarms "self", it goes to Dead stage.

C) Memory KPI (M_{KPI}): Key Performance Indicator has detected enough unsatisfied results. Memory stage is a stable stage. A Memory Key Performance Indicator has unlimited lifetime, but once it alarms "self", it goes to Dead stage.

D) Dead KPI: Key Performance Indicator has alarmed self or it is too old. Dead KPI should be eliminated.

Viable Key Performance Indicators (V_{KPIs}) are composed by T_{KPI} and MKPI:

$V_{KPI} = T_{KPI} \cup M_{KPI}$; $T_{KPI} \cap M_{KPI} = \boldsymbol{\phi}$; (Equation 61)

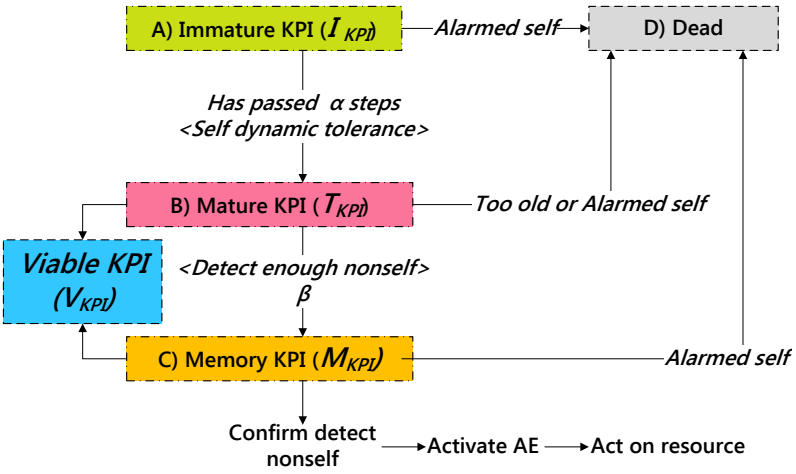


Figure 75 *Lifecycle of KPI*

Then, the Self-tolerance strategy aims at eliminating those Key Performance Indicators which made enough False Positive Errors. If a Key Performance Indicator alarms enough satisfied results as unsatisfied results, it goes to Dead stage which means it is eliminated.

Self variation strategy aims at adapting the definition of "self" (satisfied result) dynamically to fit the ever-changed business requirements.

In the following we introduce the definition for each Key Performance Indicator's lifecycle stage, evolution algorithm, immune tolerance algorithm, mature Key Performance Indicator's lifetime and dynamic memory Key Performance Indicator model.

An **Immature Key Performance Indicator** is defined as:

$I_{KPI} = \{x \mid x \in KPIs, [Self\text{-}tolerance] = non\}$. (Equation 61)

I_{KPI} has to experience a self-tolerance period. It will be eliminated if I_{KPI} alarmed self. If it survives from the self-tolerance period, it will evolve to mature Key Performance Indicator (T_{KPI}).

A **Mature Key Performance Indicator** is defined as:

$T_{KPI} = \{ x \mid x \in V_{KPI}, [Self\text{-}tolerance]=true \wedge x.KPI\_result[distance=D[0] \wedge x.count < \beta)\}$ (Equation 62)

Where x.count is the sum number of T_{KPI} alarmed non-self; β is this KPI's evolution threshold (β > 0).

When a T_{KPI} evolves to a M_{KPI}, it should detect enough non-self.

A Memory Key Performance Indicator is defined as:

$M_{KPI} = \{x \mid x \in V_{KPI}, x.KPI\_result[distance=D[0] \wedge x.count \geq \beta)\}$ (Equation 63)

Multi-cloud provides a dynamic environment. It is transforming the way computing resources are orchestrated. Definition of wastes and values

should be adapted to fit this dynamic context and to satisfy users' ever-changed requirements. As a consequence, the definition of satisfied result should be dynamically adapted to fit business requirements, the set of "self" should be adapted accordingly. To this end we define self variation:

$$\text{Self(t)}= \begin{cases} \{x_1, x_2, \dots, x_n\} & t = 0 \\ self(t-1) - self_{variation}(t) \cup self_{new}(t), t \geq 0 \end{cases}$$

(Equation 64)

$$self_{variation} = \{x | x \in self(t-1), \exists y \in V_{KPI}(t-1)(f_{check}(y,x) = 2 \wedge f_{confirm}(x) = 0)\}$$

(Equation 65)

$\text{Self}_{\text{variation}}$ is the set of mutated self-elements representing current abnormal activities.

$\text{Self}_{\text{new}}(t)$ = {y | y is the new self-element collected at time t}

(Equation 66)

$\text{Self}_{\text{new}}(t)$ is the newly defined self-elements at time t.

$$f_{check}(y,x) = \begin{cases} 2\ f_{m-list}(y.x) = 1 \wedge x \in self(t-1) \\ 1\ f_{m-list}(y,x) = 1\ \wedge x\ \notin self(t-1) \\ 0 \qquad\qquad otherwise \end{cases}$$

(Equation 67)

f_{check} (y,x)(y∈V_{KPI}, x∈ AP) is used to classify monitored execution of policy rule either identified as self or non-self. If x is matched with new self and does not belong to self(t-1), then x is sure a non-self and 1 is returned. If x is matched with new self and belongs to self(t-1), then x may be a non-self (needs to be confirmed by administrator), and 2 is returned. If x is not matched with new self, then x is identified as a self, then 0 is returned.

$$f_{confirm}(x) = \begin{cases} 1\ x\ is\ a\ self\ confirmaed\ by\ administrator \\ 0 \qquad\qquad otherwise \end{cases}$$

(Equation 68)

$$V_{KPI}(t) = M_{KPI}(t) \cup T_{KPI}(t)\ t \geq 0$$ (Equation 69)

This model is able to delete the mutated self ($\text{self}_{\text{variation}}$) in time through self-immune surveillance. Therefore, the false-negative error can be reduced. As this model can extend the depiction scope of self through adding new self (Self_{new}) into self-set. Therefore, the false-positive error can also be reduced.

To sum up the working principle of this model can be described as follow.

As this model simulates the lymphocytes growth in the marrow, an initial immature Key Performance Indicator needs to go through this self-tolerance process in a given time period evolve to a mature Key Performance Indicator.

$$I_{KPI} = \begin{cases} \{x_1, x_2, \dots, x_\xi\} & t = 0 \\ I_{tolerance}(t) - I_{maturation}(t) \cup I_{new}(t) & t \geq 1 \end{cases}$$

(Equation 70)

$I_{tolerance}(t)=\{y \mid y.distance=x.distance, y.age=x.age+1, x \in (I_{KPI}(t-1)\} - \{x \mid x \in I_{KPI}(t-1), \exists y \in self(t-1)\ f_{(m\text{-}list)}(x,y)=1\ \}$ (Equation 71)

$I_{tolerance}$ is the set of surviving I_{KPI} in $I_{KPI}(t-1)$ after one step of tolerance processes, I_{KPI} should go through α steps of tolerance processes and then evolve to T_{KPI}.

$I_{maturation}(t)=\{x \mid x \in I_{tolerance}(t), x.age>\alpha\}$ (Equation 72)

$I_{maturation}$ have undergone α steps of tolerance processes at time t.

$I_{new}(t)=\{y_1, y_2, \dots, y_\xi\}$ (Equation 73)

I_{new} is the set of new immature Key Performance Indicators generated randomly at time t.

Respecting the biological immune evolution process, we define Mature Key Performance Indicators (T_{KPIs}) associated to a fixed lifetime (λ). A Mature Key Performance Indicator (T_{KPI}) can evolve to a Memory Key Performance Indicator (M_{KPI}) when it detects enough non-self (count≥β). Otherwise, if it cannot detect enough non-self or if it detects "self", it is replaced by newly generated T_{KPI}.

$$T_{KPI}(t) \begin{cases} \phi & t = 0 \\ T'_{KPI} \cup T_{new}(t) - T_{memory}(t) - T_{dead}(t) & t \geq 1 \end{cases}$$ (Equation 74)

$T'_{KPI}(t)$: T_{KPI} undergoes one step of evolution;

$T''_{KPI}(t)$: T_{KPI} is getting older;

T_{dead} is the set of Key Performance Indicators that haven't matched enough Non-self (count ≤ β) in their lifetime(λ) or if they did false positive error at time t.

T_{clone} the clone process of mature Key Performance Indicator.

During T_{KPI}'s lifetime, the inefficient Key Performance Indicator will be killed through the clone selection processes. Efficient Key Performance Indicators will evolve to M_{KPI}.

$ap_i \in AP$ (AP is the set of all monitoring performance results including self and non-self)

$T''_{KPI}=\{y \mid y.ap=x.ap, y.age=x.age+1, y.count=x.count, x \in T_{KPI}(t-1)\}$ (Equation 75)

$T_{clone}(t)=\{y\} y.ap=x.ap,\ y.age,\ y.count = x.count + 1,\ x.count \geq \beta\}$ (Equation 76)

$T_{new}(t)=\{y \mid y.ap = x.ap,\ y.age=0,\ y.count=0,\ x \in I_{maturation}(t)\}$ (Equation 77)

$T_{memory}(t) = \{x \mid x \in T'_{KPI}(t),\ x.count \geq \beta\} \cup \{x \mid x \in T''_{KPI}(t) \wedge \exists y \in AP(t-1)\ f_{check}(x,y)= 2 \wedge f_{confirm}(y)=1\}$ (Equation 78)

T_{memory} is the set of newly generated memory Key Performance Indicators. A memory Key Performance Indicator will be deleted if it makes false-positive error which means a M_{KPI} alarms a normal activity. This dynamic model of immune memory, as well as other dynamic models discussed above can reduce both false positive error and false negative error and they can enhance the ability of self-adaptation for our governance execution system.

$$M_{KPI}(t) = \begin{cases} \phi & t = 0 \\ M_{KPI}(t-1) - M_{dead}(t) \cup T_{memory}(t) & t \geq 1 \end{cases} \quad \text{(Equation 79)}$$

In our Use Case, the Delay Rate management requirement has been used to set lots of Monitoring Policy Rules attached to the Tasks (14 rules for four Express Delivery Tasks and 20 rules for four Standard Delivery Tasks), Services (8 rules for four Express Delivry Services and 8rules for four Standard Delivery Service), Operations (8 rules for four Express Delivery Operations and 12 rules for 6 Standard Delivery Operations), Infrastructure Elements (2 rules for one Express Delivery Infrastructure Elements and 6 rules for three Standard Delivery Infrastructure Elements). Associated to these different rules, we have set Key Performance Indicators. Ajusting quality range for each Key Performance indicator is difficult and some may bring false positive/ false negative behaviors. This is why we use our Key Performance Indicator Evolution process to control the size of active Key Performance Indicator and to fit the updated "self" set.

Business Decision Makers can set different parameter values to control the KPI evolution and the size of the KPI set to fit different situations. For example, based on our use case we select a group of essential Monitoring Rules from all generated Monitoring Rules by ignoring the similar rules. We want to control the size of KPI set to only focus on the selected 20 essential different Monitoring Rules, to get a midsize governance report with precise governance results but without too much similar details. To this end, we set initial "self" n = 20, the number of newly generated immature Key Performance Indicators $\xi = 20$, and random performance records AP=60. These performance records include 10 types of Non Functional Property, such as response time, execution time, reputation, availability, etc. The performance records update period $\delta = 10$. Business Decision Makers can accelerate KPI evolution and scale up the size of KPI set by setting a lower Mature lifetime value or a lower self-tolerance step value if they focus on a global view of governance. On the other hand, they can tune the Mature lifetime value or self-tolerance value to a higher value to control the size of KPI set more strictly. As the first comparison shows in the Figure 76, we compare two sets of KPI evolution with different Self-tolernance values ($\alpha = 10$ and $\alpha = 5$), the same Mature Key Performance Indicator evolution threshold $\beta=10$ and the Mature lifetime value $\lambda = 20$ for both KPI sets. We can see that with the lower self-tolerance value ($\alpha = 5$), the KPI evolution is quicker

and the size of KPI is scaled up compare with the result from the set has higher self-tolerance value (α = 10). In our example, we set two Mature lifetime values to compare (λ =20 and λ =10) the sizes of two KPI sets, the other parameters have the same value for both KPI sets (self-tolerance value α = 10 and Mature Key Performance Indicator evolution threshold β=10). The results show that with higher Mature lifetime value, the KPI evolution is slower and the size of KPI set is smaller than the result from the KPI set with lower Mature lifetime value.

Figure 76 illustrate the sizes of KPI are adjusted by setting different Tolerance values and different Mature lifetime values, but even with different parameter values our evolution process allows the size of active Key Performance Indicator to be stable compare with the KPI set without self-evolution. We can see that without self-evolution, the size of Key Performance Indicator increases quickly and it is difficult to scale down when performance records are updated.

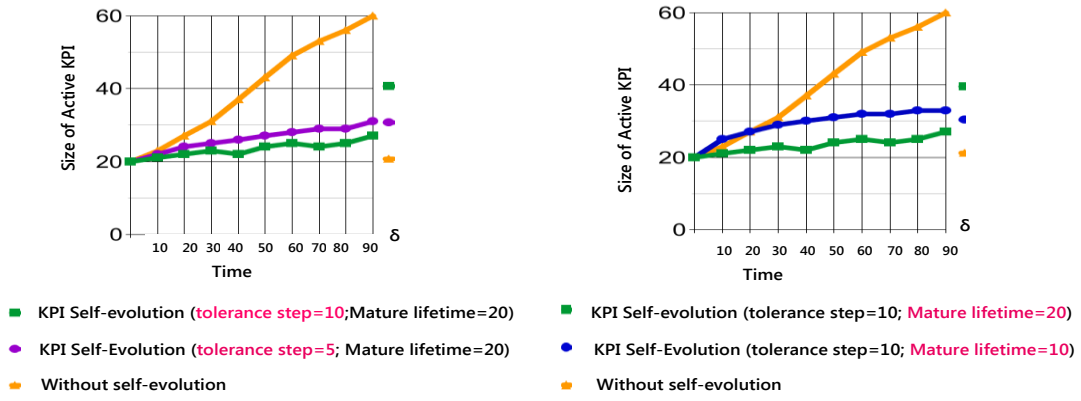


Figure 76 *Comparison of Size of Active KPI*
(Comparison 1: KPI sets have different Self-Tolerance values VS KPI set without self-evolution;
Comparison 2: KPI sets have different Mature lifetime values VS KPI set without self-evolution)

5.4.5 Use Case

We continue our Use Case demonstrating to (1) generate Computing Rules, (2) compute aggregation process.

Firstly, we parse the formalized high-level Computing Requirements. In our case, there are two registered Computing Requirements (CompReqs) in an xml file to extract root resource and root Non Functional Property. The parsing process results as following figure shows (see Figure 77).

Recorded high-level Computing Requirements:

```
<?xml version="1.0" encoding="UTF-8"?>
<ComputingRequirements version="1.0">
  <ComputingRequirement>
    <CompReq-ID>CompReq1</CompReq-ID>
    <CompReq-Goal>computing Express DistributionBP's Delay Rate in a given week</CompReq-Goal>
    <CompReq-Resource>BP_ExpressDelivery</CompReq-Resource>
    <CompReq-NFP>Delay rate</CompReq-NFP>
    <CompReq-Cxt>Express</CompReq-Cxt>
  </ComputingRequirement>
  <ComputingRequirement>
    <CompReq-ID>CompReq2</CompReq-ID>
    <CompReq-Goal>computing Standard DistributionBP's Delay Rate in a given week</CompReq-Goal>
    <CompReq-Resource>BP_StandardDelivery</CompReq-Resource>
    <CompReq-NFP>Delay rate</CompReq-NFP>
    <CompReq-Cxt>Standard</CompReq-Cxt>
  </ComputingRequirement>
</ComputingRequirements>
```

Parsing High-level Computing Requirements results:

```
Step One: Parsing High-Level Computing Requirement (CompReq)
------------------------------------------------------------
-

Total Registered High-Level CompReq number is 2
------------------------------------------------------------
-

----- Parsing Computing Requirement...

------------------------------------------------------------
Printing Parsing CompReq Results
------------------------------------------------------------
CompReq ID : CompReq1

--- Root Resource : BP_ExpressDelivery
--- Root NFP : DelayRate

------------------------------------------------------------

CompReq ID : CompReq2

--- Root Resource : BP_StandardDelivery
--- Root NFP : DelayRate

------------------------------------------------------------
```

***Figure** 77 Use Case - Parsing high-level CompReq*

After getting the Root Resource and its Non Functional Property, CompReq 1's Resource and Non Functional Property refinement processes are launched to get refined Critical Success Factor and all involved resources (see Figure 78).

Resource Refinement:

```
############################################################

Step Two ---for [ CompReq1 ]: Refine Root Resource to get Dependent resources...


-------------------- CompReq1-----------------
Start Root Resource Refinement...
total registed resource is 31

----- 1. Refining CompReq1's required Root Resource-BP_ExpressDelivery.....
BP_ExpressDeliveryhas been found.
Start Searching Dependencies ...
BP_ExpressDelivery Required Dependent Resource(s):
Rearching registered resource and dependency for BP_ExpressDelivery is finished!
!!!!!!!!!!!!!!!
----------------------------Print Refinement Results--------------------------
-

>>>>>>>>>>>>>>>>>>>> Task_E1(distribute goods from factory to depot)
>>>>>>>>>>>>>>>>>>>> Task_E2(distribute goods from depot to distribution center)

>>>>>>>>>>>>>>>>>>>> Task_E3(distribute goods from distribution center to transm
it point)
>>>>>>>>>>>>>>>>>>>> Task_E4(distribute goods from transmit point to client)
>>>>>>>>>>>>>>>>>>>> Service_1
>>>>>>>>>>>>>>>>>>>> Service_E2
>>>>>>>>>>>>>>>>>>>> Service_E3
>>>>>>>>>>>>>>>>>>>> Operation_E3-1
>>>>>>>>>>>>>>>>>>>> Operation_E3-2
>>>>>>>>>>>>>>>>>>>> Service_E4
>>>>>>>>>>>>>>>>>>>> Operation_E4-1
>>>>>>>>>>>>>>>>>>>> Operation_E4-2
>>>>>>>>>>>>>>>>>>>> Inf_E4-2
------------------------------------------------------------------------------
```

NFP Refinement Result:

```
############################################################

Step Three ---for [ CompReq1 ]: Refine Root NFP to get required CSFs...

------------------------------------------------------------

total number of registed NFP is 23
Refine [ CompReq1 ]'s Root NFP --- DelayRate
Sub-Group : subGroup-DelayRate is found!!!
searching related NFP request is accepted!

The following CSF(s) need to be invoked:

 CSF searching is finished!
-------------- Print out required CSF ----------------

>>>>>>>>>>CSF-ResponseDelayRate
>>>>>>>>>>CSF-ExecutionDelayRate
------------------------------------------------------------
```

Figure 78 *Use Case - Resource and NFP Refinement*

After completing the Resource Refinement Process, the Resource Composition Type is identified to select Aggregator-Algorithm (see Figure 79):

```
------------------------------------------------------------

Please Confirm the Composition Type of these dependency resources......
Please choose Composition Type from following Four Types:

--- Sequence(S);
--- Concurrency(C);
--- Loop(L);
--- Conditional Branching(CB);

s
confirm the Compositin Type is [ Sequence ].
```

Figure 79 *Use Case - Confirm Resource Composition Type*

Then the Generation Process of Precise Computing Requirement (PreciseCompReq) is launched. These generated PreciseCompReqs are recorded in an xml file (see Figure 80):

```
############################################################

Step Four ---for [ CompReq1 ]: Generate PreciseCompReq...

------------------------------------------------------------
[ PreciseCompReq.xml ] file is NOT existed.\nGo on to create a new PreciseCompRe
q file ...
PreciseCompReq.xml has been Created and Saved!

Total [ 2 ] PreciseCompReq(s) saved in file [ PreciseCompReq.xml ]!!!!!
```

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<PreciseComputingRequirements>
  <PreciseCompReq CompReq="CompReq1">
  <PreciseCompReq CompReq="CompReq1">
  <PreciseCompReq CompReq="CompReq2">
  <PreciseCompReq CompReq="CompReq2">
</PreciseComputingRequirements>
```

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<PreciseComputingRequirements>
 <PreciseCompReq CompReq="CompReq1">
  <PreciseCompReq-ID>PreciseCompReq-13 resources subGroup DelayRate1383848668739</PreciseCompReq-ID>
  <PreciseCompReq-Goal>To compose monitoring results...</PreciseCompReq-Goal>
  <PreciseCompReq-Resources>
   <Resource-ID>Task_E1(distribute goods from factory to depot)</Resource-ID>
   <Resource-ID>Task_E2(distribute goods from depot to distribution center)</Resource-ID>
   <Resource-ID>Task_E3(distribute goods from distribution center to transmit point)</Resource-ID>
   <Resource-ID>Task_E4(distribute goods from transmit point to client)</Resource-ID>
   <Resource-ID>Service_1</Resource-ID>
   <Resource-ID>Service_E2</Resource-ID>
   <Resource-ID>Service_E3</Resource-ID>
   <Resource-ID>Operation_E3-1</Resource-ID>
   <Resource-ID>Operation_E3-2</Resource-ID>
   <Resource-ID>Service_E4</Resource-ID>
   <Resource-ID>Operation_E4-1</Resource-ID>
   <Resource-ID>Operation_E4-2</Resource-ID>
   <Resource-ID>Inf_E4-2</Resource-ID>
  </PreciseCompReq-Resources>
  <PreciseCompReq-CompositionType>Sequence</PreciseCompReq-CompositionType>
  <PreciseCompReq-CSF>CSF-ResponseDelayRate</PreciseCompReq-CSF>
 </PreciseCompReq>
```

Figure 80 *Use Case- Precise Computing Requirement Generation Result*

Generating Computing Rules from Precise Computing Requirement (Precise-CompReq) requires appropriate Aggregator-Algorithm. The convenient Aggregator selection requires to match the Resource Composition Type and Critical Success Factor with Aggregators' attribute (CompositionType and Critical Success Factor). Following figure shows the results of extracting Precise-CompReq's Resource Composition Type and Critical Success Factor by parsing generated Precise-CompReq file (see Figure 81).

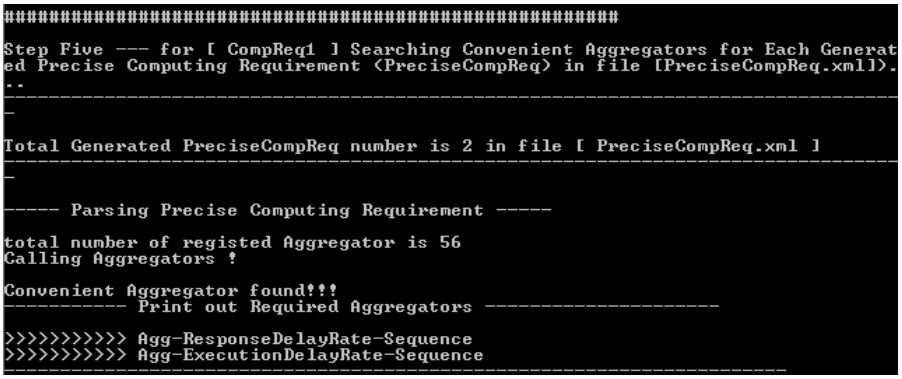


Figure 81 *Use Case - Parsing Generated Precise-CompReq to select convenient Aggregator*

After selecting the appropriate Aggregator, the Computing Rules are generated. These generated Computing Rules are recorded into an xml file for further use (see Figure 82 and Figure 83):

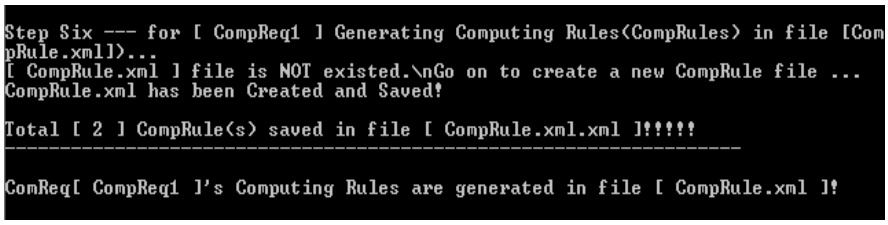


Figure 82 *Use Case - CompRule Generation Result*

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<ComputingRules>
  <CompRule CompReq="CompReq1">
  <CompRule CompReq="CompReq1">
  <CompRule CompReq="CompReq2">
  <CompRule CompReq="CompReq2">
</ComputingRules>
```

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<ComputingRules>
 <CompRule CompReq="CompReq1">
  <CompRule-ID>CompRule-13 resources subGroup-DelayRate1383848771900</CompRule-ID>
  <CompRule-Goal>To compose monitoring results... </CompRule-Goal>
  <CompRule-Resources>
   <Resource-ID>Task_E1(distribute goods from factory to depot)</Resource-ID>
   <Resource-ID>Task_E2(distribute goods from depot to distribution center)</Resource-ID>
   <Resource-ID>Task_E3(distribute goods from distribution center to transmit point)</Resource-ID>
   <Resource-ID>Task_E4(distribute goods from transmit point to client)</Resource-ID>
   <Resource-ID>Service_1</Resource-ID>
   <Resource-ID>Service_E2</Resource-ID>
   <Resource-ID>Service_E3</Resource-ID>
   <Resource-ID>Operation_E3-1</Resource-ID>
   <Resource-ID>Operation_E3-2</Resource-ID>
   <Resource-ID>Service_E4</Resource-ID>
   <Resource-ID>Operation_E4-1</Resource-ID>
   <Resource-ID>Operation_E4-2</Resource-ID>
   <Resource-ID>Inf_E4-2</Resource-ID>
  </CompRule-Resources>
  <CompRule-CompositionType>Sequence</CompRule-CompositionType>
  <CompRule-CSF>CSF-ResponseDelayRate</CompRule-CSF>
  <CompRule-Agg>Agg-ResponseDelayRate-Sequence</CompRule-Agg>
  <CompRule-Agl>ResponseDelayRate.value = [1-{product{unit.value i...n}}]*100%</CompRule-Agl>
 </CompRule>
```

Figure 83 *Use Case - Generated CompRule*

At the same time, the log file < CompRule Generation Log> is generated (see Figure 84).

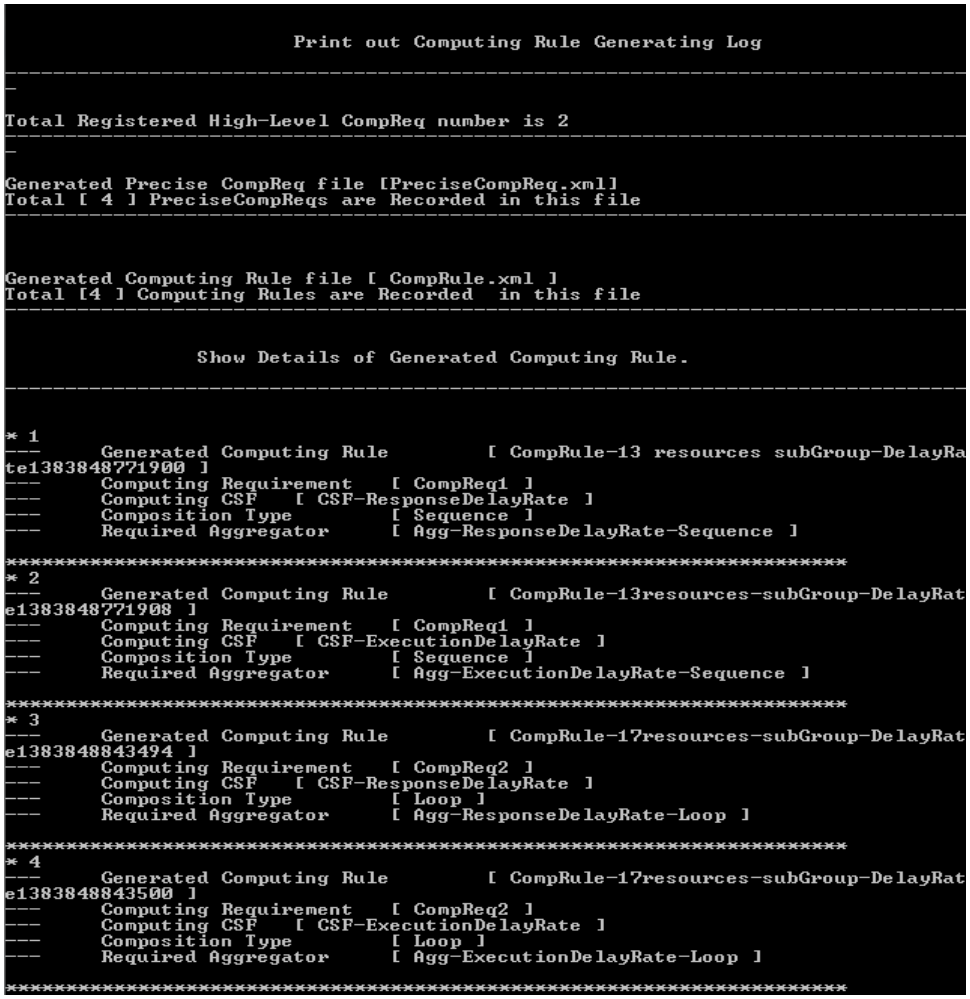


Figure 84 *Use Case - CompRule Generation Log*

At the end of this process, the following computing rules are generated:

*- CompRule 1= (ID= "CompRule 1", Goal = "compute Distribution Express Business Process' Response Delay Rate", CompRule-Data = "Express Delivery Business Process required Resources' Response Delay Rate Monitoring Results (13 Resources)", CompositionType ="Sequence", Algorithm =: "[1- ($\prod_{i=1}^{n}$ DelayRate i)]*100%", Aggregator="Agg-ResponseDelayRate-Sequence");*

- CompRule 2= (ID= "CompRule 2", Goal="compute Distribution Business Process' Execution Delay Rate", CompRule-Data = "Express Delivery Business Process required Resources' Response Delay Rate Monitoring Results (13 Resources)", CompositionType ="Sequence", Algorithm =" [1-

*($\prod_{i=1}^{n} DelayRate\ i$)]*100%", Aggregator="Agg-ExecutionDelayRate-Sequence");*

*- CompRule 3= (ID= "CompRule 3",Goal="computing Standard Business Process' Response Delay Rate", CompRule-Data = "Standard Delivery Business Process required Resources' Response Delay Rate Monitoring Results (17 Resources)", CompositionType ="Loop", Algorithm ="[1- ($\prod_{i=1}^{n} DelayRate\ i$)]*100%, Delay Rate = (delayed time/measured time)*100%", Aggregator="Agg-ResponseDelayRate-Loop");*

*- CompRule 4= (ID= "CompRule 4", Goal="computing Standard Delivery Business Process' Execution Delay Rate", CompRule-Data = "Standard Delivery Business Process required Resources' Response Delay Rate Monitoring Results (17 Resources)", CompositionType ="Loop", Algorithm ="[1- ($\prod_{i=1}^{n} DelayRate\ i$)]*100%, Delay Rate = (delayed time/measured time)*100%", Aggregator="Agg-ExecutionDelayRate-Loop");*

These computing rules can invoke the convenient aggregators to select required Key Performance Indicators' monitoring results according to the aggregation algorithms to compute comprehensive governance results.

a) Aggregation of Service's delay rate (Express Delivery Business Process):

- Resources' Composition (see Figure 85):

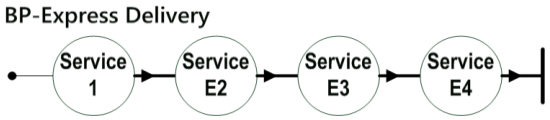


Figure 85 *Services' Composition for Express Delivery BP*

- Aggregation Algorithm: [1-($\prod_{i=1}^{n} DelayRate\ i$)]*100%
(Equation 80)

- Aggregated Result:

Express Delivery's Response Delay Rate= 1-(0.988*0.989*0.99*0.98) = 0.052 (Equation 81)

Express Delivery's Execution Delay Rate= 1-(0.987*0.982*0.978*0.979) = 0.072 (Equation 82)

b) Aggregation of Operations' Delay for Business Process Express Delivery:

- Resources' Composition (see Figure 86):

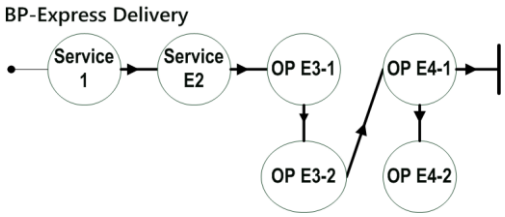


Figure 86 *Operations' Composition for Task 1*

- Aggregation Algorithms for governance details:

Completed the Business Process Express Delivery with maximum operations;

Algorithm: [1- ($\prod_{i=1}^{n} DelayRate\ i$)]*100% (Equation 83)

Aggregated results:

Response Delay Rate: 1-(0.991*0.992*0.99*0.997*0.993*0.988) = 0.048; (Equation 84)

Execution Delay Rate: 1-(0.989*0,987*0.989*0.989*1*0.985) = 0.060. (Equation 85)

All of these aggregation results can be presented into a mashup dashboard (which presents these results to WSO2 BAM's dashboard) (see Figure 87).

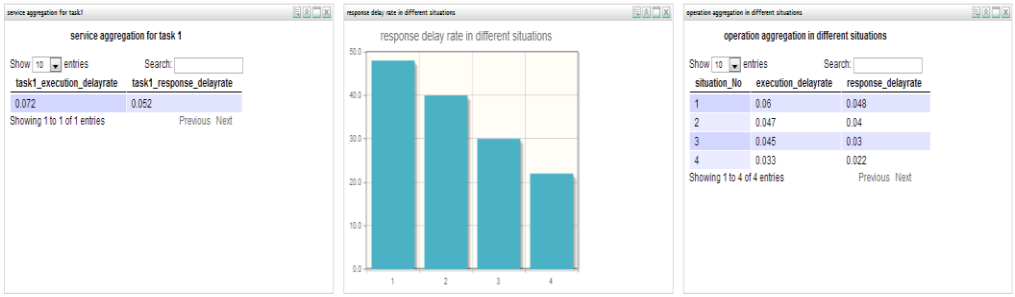


Figure 87 *Use Case - Aggregation Results Presentation*

5.5 Conclusion

In this chapter we present our contribution to resolve the two remainingresearch questions:

Q3: How can we make the governance approach be customized and self-adjusted to meet different governance requirements?

Q4: How governance and adapting processes can be automatic and benefit business outcomes?

To this end, we propose a Governance Execution and Adapting Framework. This framework aims at governing the performance of business processes and quality of all resources according to customized governance requirements. It allows all activities in business process workflow to create values for Business Decision Maker and to minimize wastes in Business Decision Maker required business process. Continuing the previously introduced Preparation Phase of our Governance Loop (in chapter 4), the processes of this Governance Execution and Adapting Framework implement the last two phases of our Governance Loop: Governance Execution Phase and Adapting Phase.

This Multi-layer Governance Framework includes two main models:

1) Multi-layer Monitoring Model which aims at implementing customized governance requirements at runtime and monitoring quality of resources the performance of business processes. In this model, we proposed a monitoring requirement formalization process to set a generic monitoring process fitting various requirements. Then we proposed a monitoring policy rule generation process which takes advantage of both Model-Driven Engineering and of Pattern-based Engineering approach to generate customized monitoring policy rules. These generated monitoring policy rules invoke Key Performance Indicators (KPIs) to implement each monitoring policy rule.

2) Multi-layer Computing Model. This model aims at collecting and computing initial runtime monitoring results, giving comprehensive governance results and making monitoring to be beneficial to business. To implement the computing process, we propose an automatic computing rule generation process according to customized computing requirements. Generated Computing Rules can invoke convenient composition algorithms to aggregate related resources/ Non Functional Properties real-time monitoring results into required comprehensive results and present on dashboards. These aggregation algorithms are designed according to the composition of involved resources and the characteristics of Non Functional Properties.

In addition, we design an Immunity Inspired autonomic management strategy for Key Performance Indicator's evolution and lifecycle management to make governance process be dynamic and autonomic. To make self-driven correction mechanism, Action Engines (AEs) can be invoked by Key Performance Indicators automatically according to Key Performance Indicators' monitoring result. Furthermore, a Use Case has been used to demonstrate the application of this governance framework.

6 Conclusion

Maintaining business competitiveness and achieving agility of Business and IT is an important motivation for organizations to adapt their businesses into the ever-changing environment. The rapid rise of Cloud Computing is enabling a wide array of new business models and these new models have greatly changed the way we organize and provide business resources. However, even cloud computing has emerged as an important solution offering enterprises a potentially cost effective model to ease their computing needs and accomplish business objectives. This loosely coupled paradigm also brings many uncertainties and open challenges, such as controlling quality of service, managing performance of business process, optimizing resources' utilization and managing violation of agreements, etc. These challenges are hindering the development of Cloud Computing and making businesses are under increasing pressure to sharpen their business practices. It is also impacting the use of emerging IT. To solve these problems, a comprehensive resource governance and management approach is required to understand the real-time status of organizations from business to infrastructure elements, thus guarantee organizations gain promised benefits from emerging business paradigms and technology.

In this research work, we proposed a Service-Oriented Multi-layer Management and Governance architecture (SO-MGA).

After raising and analyzing our main research question, we divide this main question into sub-questions, and searched for existing solutions for these sub-questions. To overcome the limits of existing solutions, SO-MGA aims at narrowing the gap between business and IT, at providing a customized multi-layer governance solution to simplify the management and governance from business perspective. This will also guarantee that involved resources' activities add value on business side, as well as make sure that the technical adjustments always meet the business needs. Our solution pays attention on monitoring business performance and quality of resources' Non Functional Properties at runtime. We also design monitoring composition algorithms to provide comprehensive governance results. Besides, we design autonomic management strategy to implement our management and governance without interfering with organizations' business processes. A Use Case is introduced to demonstrate and evaluate our solution gradually.

6.1 Work Summary

In chapter 1, we analyzed the research background. The new business paradigms and information technologies have dramatically changed the way business is run. These emerging business paradigms and technologies bring new opportunities but also new challenges. According to our research a lack of a comprehensive and customizable governance and monitoring solution considering business requirements in cloud environment is one the biggest obstacles for organizations to get benefits from service-oriented computing and cloud computing. Therefore, how can we achieve a customizable governance solution to monitor and control quality of services, performance of business, and guarantee organizations obtain promised benefits from cloud? We divided this generic question into 4 sub-questions: 1) Understand clearly what should be governed? 2) Define governance objects. This needs to analyze how these objects impact on organizations' performance and their business outcomes. 3) Design a customized governance solution to fit the requirements dynamically at runtime without interfere organizations' ongoing business processes. 4) Make organizations obtain benefits from our governance solution to achieve organizations' business and IT agility.

In chapter 2 (the State of The Art), we firstly introduced the global context. The development of IT/IS, Information Systems (ISs) play a vital role for organizations. Besides traditional product value chain, organizations pay more and more attention on information value chain. IS agility is the key factor for business agility, more and more organizations adapt Lean Think to their management to achieve IS and business agility. The most important step for lean implementation is identification of waste before removing them. In our IT collaborate business environment, ultimate essential concern is quality of information which has to fit the business requirements and add value to business processes. Delivering tangible business value, business processes need to swiftly adapt its strategies to reflect IT changes, business processes and ISs should be aligned to achieve organization's agility.

According to this analysis, we can answer our first sub-question: the elements of information value chain are the most important governance objects for our research. Then for answering the second sub-question, we need to understand the interplay between governance objects and how these information value chain's elements impact organizations' business performance. We introduced Business Process Management (BPM) which provides understanding and metrics for business context. Enterprise Architecture (EA) gives a platform for BPM to implement enterprise's business strategy. Combining EA and BPM makes EA gain additional benefits from BPM, such as optimize and deploy process models for maximized business outcomes, consider BPM into IT reuse and governance, etc. After understanding the close connection between IT and

BPM, we analyze the top four EA frameworks: Zachman, TOGAF, FEA and Gartner, each of them have their own strengths and weakness. However, none of them provide a clear Success Measurement and Governance Guidance. Therefore, it requires a new solution which can develop the disciplines of EA. The requirement of combing BPM with EA fits the ability of SOA. Furthermore, the value proposition of SOA is centered on agile and aligned business and IT design and delivery. Layers of SOA simplify the complexity from business perspective the way Non-functional properties (Non Functional Properties) and Service Level Agreement (SLA) constrain the way services' functional properties are achieved. However, SLAs do not pay much attention on the higher-level aspects of interaction between business and service-based applications. Therefore, the Business-Level Agreement (BLA) becomes a critical concern for enterprises to maintain the long-term business value of using these service-based applications. BLA is complementary to the technical SLAs. Performing these agreements (SLAs /BLAs), requires SOA governance to define the set of policies, rules, and enforcement mechanisms for developing, using, and evolving service-oriented systems and for analyzing their business value. After analyzing several existing SOA governance solutions, we list some challenges for SOA governance, such as delivery value to stakeholders, compliance to standards, and dynamicity from consumer perspective.

The third and fourth sub-questions (i.e. building a customized and dynamic governance solution) lead to a new requirement, namely performance measurement to fit the service-oriented environment. We analyze existing performance measurement solutions and their limits. As they consumed a lot of manpower to develop measurement system, they are not adaptable for updating. They do not easily meet other requirements, such as fitting the service-oriented environment. In addition, we compared the traditional dashboards and scorecards. They have their own features (dashboards show runtime detailed status, while scorecards show periodic snapshots summary status). In order to make the governance customizable and fit the service-oriented environment, we list some challenges to provide customizable presentation (chose right indicators to show the most important aspects of business performance, consider showing results good for further business decisions). Due to the complexity of organizations, organizing required Key Performance Indicators (KPIs) automatically requires an autonomic management solution. We adapt the immunologically inspired theory to our governance solution to management our governance elements automatically. Considering the layers (Software as a Service: SaaS, Platform as a Service: PaaS, Infrastructure as a Service: IaaS) and deploying models (Private Cloud, Community Cloud, Public Cloud and Hybrid Cloud) of Cloud Computing, it involves more complex monitoring systems, which have to be robust, scalable and fast, to be able to manage and

verify a large number of resources. We analyzed some open issues for Cloud Computing Governance, such cross-layer, cross-domain governance, monitoring novel network architecture, etc.,

To overcome the existing solutions' limits, we introduce our solution: Service-Oriented Multi-layer Management and Governance Architecture (SO-MGA). In the chapter 3, we globally introduce our multi-layer management and governance architecture. Considering the layers of Cloud and combining business process with information value chain, we extend the traditional XaaS to Business as a Service (BaaS) layer and defined 3 "horizontal" layers (viz. Business Process Layer, Business Service Layer and Business Implementation Layer) according to the functional features of resources, a "vertical" governance layer and a top Business decision layer. Achieving the simplicity and dynamicity, we designed an Integrated Management and Governance Bus as a middleware to exchange messages and implement interactions between components. In order to give a global view of our governance solution, we also introduce our governance architecture's working principle and a general definition of resource and classification of Non Functional Properties.

In chapter 4 we detail our Multi-Layer BaaS Management and Governance Preparation Framework which includes BaaS Resource Organization Model and BaaS Management Negotiation and Governance Preparation Model. In the Resource Organization Model section, we explain the deconstruction and formalization of Business Decision Maker's management requirement, selection and organization of required resources. The Negotiation and Governance Preparation Model includes the definition of our Multi-level Agreements. The Multi-level Agreements (MLAs) include Business Process Level Agreement (BPLA) and Business Service Level Agreement (BSLA). This MLA aims at working as a mediator between Business Decision Maker and Service Providers to reduce the violation and number of reconfiguration for Business Decision Maker, as well as making technical adjustments always meet business requirements. A Logistics Company Use Case is used to show Multi-layer BaaS Management Framework can be used.

In chapter 5, we introduce our Multi-Layer BaaS Governance Framework which includes a Multi-layer Monitoring Model (runtime monitoring) and a Multi-layer Computing Model (composing initial monitoring results to comprehensive governance results). To achieve this customizable governance, we designed a requirement formalization process and a governance rule generation process. Furthermore, achieving the autonomic management, we designed an immunity inspired solution to manage governance elements automatically. Improving organization's performance according to governance results, we designed a composition algorithms considering Non Functional Properties' feature and governance objects' interaction relationships. The

Logistics Company Use Case is used to demonstrate how Multi-layer BaaS Governance Framework can be used.

In a summary, our Service-Oriented Multi-layer Governance Architecture aims at providing a solution to implement the agility of business and IS. This policy-based governance solution focuses on the performance of both business value chain and information value chain, and uses a formal model. It can formalize governance requirements, generate customized governance rules, and deploy governance elements such as Key Performance Indicators and Aggregators. Monitoring implementation allows composing initial results as comprehensive results on dashboards. These governance results aim not only at providing real-time status of business resources' performance but also at providing periodic analysis for both information and business value chain. According monitoring results we can adjust the performance of resource. Moreover, we design an autonomic management to manage Key Performance Indicator's evolution and lifecycle to improve the dynamicity of our governance without any interference.

6.2 Future Work

Based on our work, following research topics could extend it in the future.

- Optimize Selection of Resources

In this work we have proposed an approach to select and manage Non Functional Property-based resources for business requirement. However, in today's cloudy environment, how to select the most appropriate resources from the massive resource pool efficiently is still a challenge for business decision makers. It would be interesting to explore universal approaches that support extracting features of resources and sort resources according to specific needs, to accelerate resources' update and to improve the efficiency of resource management and selection.

- Enhance the Efficiency of Agreements' Management

The collaboration of internal and external enterprises is increasing in cloud computing environment. With the XaaS concept, an enterprise or an organization could cooperate with difference Service Providers. To constrain mutual obligation and responsibility there could have different contracts and agreements between with these various Service Providers. A standard approach to manage all of these contracts and agreements and minimize the harm from violations is important for achieving business agility.

- Catalogue Non Functional Properties and Standardize Non Functional Property's Aggregation Algorithm

It would be interesting to explore an efficient approach to catalogue Non Functional Properties in various domains and give specific definition to Non Functional Property for particular usages. As we have introduced in this dissertation, some Non Functional Properties cannot be calculated but we still need to aggregate difference resources' Non Functional Properties to have a comprehensive view. This is why a standard approach to aggregate Non Functional Properties in various domains is important for managing and governing resources.

- Analyze and Predict Governance Results

Governance system will provide massive data. It is important to explore an approach to analyze this big data to extract meaningful information according to particular business requirement. Furthermore, it would be interesting to explore an approach to predict the future situation by analyzing recorded history data. The prediction could assist enterprises to orchestrate their resource more reasonably.

Bibliography

[Aarabi 2011] **Aarabi M., Kashefi M., Khoei M.R.** , Verification and Validation in GERAM Framework for Modeling of Information Systems, in: International Journal of Scientific and Engineering Research (IJSER), Vol. 2, Issue 10, October 2011, pp.1-10

[Aburub et al. 2007] **Aburub, F, Odeh, M, and Beeson, I,**, Modelling non functional requirements of business processes, Information and Software Technology, Vol. 49, No. 11-12, P. 1162-1171, 2007

[Aceto et al. 2013] **Giuseppe Aceto, Alessio Botta, Walter de Donato, Antonio Pescapè**, Cloud monitoring: A survey, Computer Networks, Volume 57, Issue 9, 19 June 2013, Pages 2093-2115,

[Aedo et al. 2010] **Aedo, I., Diaz, P., Carroll, J. M., Convertino, G., & Rosson, M. B.** (2010). End-user oriented strategies to facilitate multi-organizational adoption of emergency management information systems. Information Processing and Management, 46(1), 11–21.

[Alhamad et al., 2011] **Alhamad, M., Dillon, T., Chang, E.,** A survey on SLA and performance measurement in cloud computing, on the move to meaningful internet systems: OTM 201, Springer Berlin, 7045, 2011., pp. 469-477.

[Alukal 2003] **Alukal G.**, Create a lean mean machine, Quality Progress, Vol. 36, No. 4, 2003, pp.29-35

[Alonso et al. 2004] **Alonso, G., Casati, F., Kuno, H., Machiraju, V**.: Web Services: Concepts, Architectures and Applications. Springer, Heidelberg (2004)

[Al-Kilidar et al., 2005] **H. Al-Kilidar, K. Cox, B. Kitchenham**, The use and usefulness of the ISO/IEC 9126 quality standard, in: Proceedings of the International Symposium on, Empirical Software Engineering, 2005, pp. 122–128.

[Arsanjani 2004] **Ali Arsanjani**, Service-Oriented modeling and architecture,2004, http://www.ibm.com/developerworks/library/ws-soa-design1/

[Armbrust et al. 2010] **Michael Armbrust, Armando Fox, Rean Griffith**, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. 2010. A view of cloud computing. Commun. ACM 53, 4 (April 2010), 50-58.

[Aydin et al. 2010] **Ilhan Aydin, Mehmet Karakose, Erhan Akin**, Artificial immune classifier with swarm learning, Engineering Applications of Artificial Intelligence, Volume 23, Issue 8, December 2010, Pages 1291-1302.

[Bajwa et al. 2008] **Bajwa I. S., Rafaqut K., Shahzad M., Shahid N., M. Abbas C.,** [2008] "SOA and BPM Partnership: A paradigm for Dynamic and Flexible Process and I.T. Management", Proc. of World Academy of Science, Engineering and Technology Vol. 35, No.4 , pp 16-22, Nov. 2008

[Bailey and Francis 2008] **Kate Bailey, Mark Francis,** Managing information flows for improved value chain performance, International Journal of Production Economics, Volume 111, Issue 1, January 2008, Pages 2-12

[Bauch 2004] **BAUCH, C.,** Lean Product Development enabling display: Making Waste Transparent, Munich: Technical University of Munich, diploma thesis (2004)

[Bajwa 2011] **Bajwa, I. S.,** SOA Embedded in BPM: A High Level View of Ob-ject Oriented Paradigm, WASET 2011 , pp. 304-308 , 2011

[Banuelas et al. 2006] **Banuelas, R., Tennant, C., Tuersley, I., Tang, S.,** (2006) "Selection of six sigma projects in the UK", The TQM Magazine, vol.18, No.5, pp.514 – 527

[Basu 2012] **Basu M.,** Artificial immune system for combined heat and power economic dispatch, International Journal of Electrical Power & Energy Systems, Volume 43, Issue 1, December 2012, pp.1-5

[Becker et al. 2004] **Becker, J., Kugeler, M., and Rosemann, M.**, Process Management: A Guide for the Design of Business Processes, Berlin. Germany. Springer, 2004.

[Bertolino and Polini 2009] **Bertolino, A.,; Polini, A.,** "SOA Test Governance: Enabling Service Integration Testing across Organization and Technology Borders," Software Testing, Verification and Validation Workshops, 2009. ICSTW '09. International Conference on, pp.277,286, 1-4 April 2009

[Bernhardt and Detlef 2008] **Bernhardt J., and Detlef Seese D.**: A conceptual framework for the governance of service-oriented architectures. In George Feuerlicht and Winfried Lamersdorf, editors, Service-Oriented Computing " ICSOC 2008 Workshops, volume 5472 of Lecture Notes in Computer Science, pp. 327–338. Springer Berlin /Heidelberg, 2008.

[Beuren, et al. 2013] **Fernanda Hänsch Beuren, Marcelo Gitirana Gomes Ferreira, Paulo A. Cauchick Miguel**, Product-service systems: a literature review on integrated products and services, Journal of Cleaner Production, Volume 47, May 2013, Pages 222-231,

[Berrah, 2012] **Lamia Berrah , François Vernadat** , Towards A System-Based Model For Overall Performance Evaluation In A Supply Chain Context . in The Open Industrial & Manufacturing Engineering Journal 5, 1874-1525 (2012) 8-18

[Bieberstein et al. 2005] **Bieberstein, N., Bose, S., Walker, L., Lynch, A.,** 2005. Impact of service-oriented architecture on enterprise systems, organizational structures, and individuals. IBM Systems Journal 44 (4), 691–708.

[Bianco et al. 2008] **P. Bianco, G.A. Lewis, and P. Merson**, "Service Level Agreements in Service-Oriented Architecture Environments," TECHNICAL NOTE CMU/SEI- 2008-TN-021, 2008.

[Bourne et al. 2000] **Bourne, M., Mills, J., Wilcox, M., Neely, A., & Platts, K**. (2000). Designing, implementing and updating performance measurement systems. International Journal of Operations and Production Management, 20(7), 754–771.

[Boukerche et al. 2007] **Boukerche A, Machado R B, Juca K R L, Sobral J B M, NotareMSMA,** "An agent based and biological inspired real -time intrusion detection and securitymodel for computer network operations", International Journal of Computer Communications, Vol.30, pp(s):2649–60, 2007.

[Brynjolfsson and Hitt 2003] **E. Brynjolfsson, LM. Hitt**, Computing productivity: Firm-level evidence, Review of economics and statistics, Vol. 85, No. 4,pp. 793-808, 2003

[Brandley 1996] **Bradley P.** , A performance measurement approach to the reengineering of manufacturing enterprises, Ph.D. Thesis, CIMRU, NUI Galway, Ireland, 1996

[Braun and Winter, 2007]**Christian Braun and Robert Winter**. 2007. Integration of IT service management into enterprise architecture. In Proceedings of the 2007 ACM symposium on Applied computing (SAC '07). ACM, New York, NY, USA,

[Brewer and Speh 2001] **P. Brewer, T. Speh**, Adapting the balanced scorecard to supply chain management, Supply Chain Management Review (March–April) (2001) 48–56.

[Bruce 2006] **Bruce S.** , "BPM on SOA: What would it look like? – Part 1 ", Bruce Silver Associates Article, August 21, 2006

[Buyya et al. 2009] **Buyya, R., Yeo, S.C., Venugopal, S., Broberg, J., Brandic, I.**, Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility, Future Generation Computer Systems, Volume 25, Issue 6, June 2009, Pages 599-616,

[Canter 2000] **Canter, J.,** An agility-based OODA model for the e-commerce/e-business enterprise. (2000) Retrieved from: http://www.belisarius.com/modern_business_strategy/canter/canter.htm.

[Castro and Zuben 2002] **de Castro and Von Zuben**, 2002. Learning and optimization using the clonal selection principles. IEEE Trans. Evolut. Comput. v6 i3. 239-251.

[Cavalieri et al. 2012] **Sergio Cavalieri, Giuditta Pezzotta**, Product–Service Systems Engineering: State of the art and research challenges, Computers in Industry, Volume 63, Issue 4, May 2012, Pages 278-288,

[Conboy and Fitzgerald 2004] **Conboy, K. B., Fitzgerald**. Towards a conceptual framework of agile methods: A study of agility in different disciplines. In Proceedings of the 2004 ACM work-shop on inter disciplinary software engineering research. New-port Beach, CA, (2004) pp. 37–44.

[Clayman et al., 2010] **Clayman, S., Galis, A., Chapman, C., Toffetti, G., Rodero-Merino, L., Vaquero, L. M., Nagin, K., Rochwerger B.**, Monitoring Service Clouds in the Future Internet. In: Tselentis, G and Galis, A and Gavras, A and Krco, S and Lotz, V and Simperl, E and Stiller, B and Zahariadis, T, (eds.) Towards the Future Internet - Emerging Trends from European Research., 2010. pp. 115 -126

[Choe 2008] **Jong-min Choe,** Inter-organizational relationships and the flow of information through value chains, Information & Management, Volume 45, Issue 7, November 2008, Pages 444-450

[Cho, et al., 2012] **Dong Won Cho, Young Hae Lee, Sung Hwa Ahn, Min Kyu Hwang**, A framework for measuring the performance of service supply chain management, Computers & Industrial Engineering, Volume 62, Issue 3, April 2012, Pages 801-818,

[Chen and Aickelin 2004] **Chen, Q. and Aickelin, U.** 2004. Movie recommendation systems using an artificial immune system. In Proc. of ACDM 2004. UK

[Chen 2010] **Bo Chen,** Agent-based artificial immune system approach for adaptive damage detection in monitoring networks, Journal of Network and Computer Applications, Volume 33, Issue 6, November 2010, Pages 633-645, 2010.03.011.

[Chen et al., 1997] **D. Chen, B. Vallespir, G. Doumeingts,** GRAI integrated methodology and its mapping onto generic enterprise reference architecture and methodology, in Computers in Industry, Volume 33, Issues 2–3, September 1997, pp.387-394

[Chesbrough and Garman 2009] **Chesbrough, H. W., and Garman, A. R.** (2009). How open innovation can help you cope in lean times. Harvard Business Review, 87(12), 68–76.

[Chung, 2009] **L. Chung, J. C. Leite**, "On Non-Functional Requirements in Software Engineering," in Conceptual Modeling: Foundations and Applications, Springer-Verlag, Vol. 5600, 2009, pp 363-379.

[Cusumano and Nobeoka 1998] **Michael A. Cusumano, Kentarō Nobeoka**, Thinking beyond lean: how multi-project management is transforming product development at Toyota and other companies, Free Press, 1998, p248.

[Cuervo et al. 2010] **E. Cuervo, A. Balasubramanian, D. Cho, A.Wolman, S. Saroiu, R. Chandra, P. Bahl, Maui**: making smartphones last longer with code offload, in: Proceedings of the 8th international conference on Mobile systems, applications, and services, ACM, 2010, pp.49–62.

[Crandall and Coffey 2005] **R. Crandall, B. Coffey**. Sailing the seven C's Industrial Engineer, 37 (8) (2005), pp. 35–39

[Crute et al. 2003] **Crute,V., Ward, Y., Brown, S., Graves, A.**, Implementing lean in aerospace: Challenging the assumptions and understanding the challenges Technovation, 23 (12) (2003), pp. 917–928

[Collen 2006] **Colleen F.,** "Special Report: BPM Inside the Belly of the SOA Whale", Web Services News, June 15, pp. 1-4, 2006

[Davenport 1993] **Thomas H. Davenport**, Process Innovation: Reengineering work through information technology, Harvard Business Press, 1993, p337

[Dasgupta et al. 2005] **Dasgupta D, Yu S, Majumdar N S,** "MILA—multi level immune learning algorithm and its application to anomaly detection". Soft Computing, Vol. 9, No. 3, pp(s):172–84, 2005.

[Derler and Weinreich 2007] **Derler, P., and Weinreich, R**. 2007. Models and tools for SOA governance, In: Draheim, D., Weber, G. (Eds.),

Trends in Enterprise Application Architecture. Berlin: Springer, Vol. 4473, pp. 112–126.

[Doerr et al. 2005] **J. Doerr, D. Kerkow, T. Koenig, T. Olsson, T. Suzuki**, Non-functional requirements in industry – three case studies adopting an experience-based NFR method, in: Proceedings of the 13th IEEE International Conference on Requirements Engineering, RE'05, Paris, France, 2005, pp. 373–382.

[Doumeingts 1995] **G. Doumeingts, F. Clave, Y. Ducq, ECOGRAI**, A method to design and to implement Performance Measurement Systems for industrial organizations — Concepts and application to the Maintenance function, in Benchmarking — Theory and Practice IFIP Advances in Information and Communication Technology 1995, pp 350-368

[EAO 2004] **Enterprise Architecture**: Overview and Uses, US Department of De-fense, http://www.tricare.osd.mil/Architecture/MHS-EAOVERVIEW-and-uses-20021119.htm, accessed in 2004

[Elbashir et al. 2008] **M. Elbashir, P. Collier, M.J. Davern**, Measuring the effects of business intelligence systems: The relationship between business process and organizational performance, Inter-national Journal of Accounting Information Systems, Volume 9, Issue 3, September 2008, Pages 135-153,

[Easton 2012] **George S. Easton, Eve D. Rosenzweig**, The role of experience in six sigma project success: An empirical analysis of improve-ment projects, Journal of Operations Management, Volume 30, Issues 7–8, November 2012, Pages 481-493

[Eenoo et al. 2005] **Eenoo, C. V., Hylooz, O., and Khan, K. M.** (2005). Ad-dressing Non-Functional Properties in Software Architecture using ADL. In Proceedings of the 6th Australian Workshop on Software and Systems Architectures - AWSA'05, Brisbane, Australia, pages 6–13, March 29, 2005.

[Eckerson 2005] **Eckerson W.,** Performance Dashboards: Measuring, Monitoring, and Managing Your Business, John Wiley & Sons, 2005,

[Erl 2007] **Erl, Thomas**, SOA – Principles of Service Design. Boston: Prentice Hall, 2007

[Fawcett and Magnan 2008] **Fawcett, S. E., Magnan, G. M., McCarter, M. W.:** Benefits, barriers, and bridges to effective supply chain management. J. Supply Chain Management. vol. 13(1). pp. 35–48. (2008).

[FEA 2005] **FEA** Consolidated Reference Model Document. www.whitehouse.gov May 2005.

[FEA 2007] **FEA:** Federal Enterprise Architecture Program Management Office (2007). FEA Practice Guidance

[FEA 2009] **FEA:** Improving Agency Performance Using Information and Information Technology, 2009, p40. http://www.whitehouse.gov/sites/default/files/omb/assets/fea_docs/

[Folan and Browne 2005] **Folan, P., Jim Browne, J.,** A review of performance measurement: Towards performance management, Computers in Industry, Vol. 56, No. 7, pp. 663-680, (2005)

[Franco-Santos et al. 2007] **Franco-Santos, M., Kennerley, M., Micheli, P., Martinez, V., Mason, S., Marr, B., Gray, D., Neely, A.,** (2007) "Towards a definition of a business performance measurement system", International Journal of Operations & Production Management, Vol. 27 Iss: 8, pp.784 – 801

[Garrison et al. 2012] **Garrison, G., Kim, S., Wakefield, R. L.**, Success factors for deploying cloud computing. Commun. ACM 55(9), 2012, pp., 62-68.

[Ganguly and Deutsch, 2004] **Niloy Ganguly, Andreas Deutsch,** Developing Efficient Search Algorithm for P2P networks using proliferation and mutation, In Third International Conference, ICARIS 2004 Proceedings, Catania, Sicily, Italy, September 13-16, 2004. pp. 357-371

[Gogouvitis et al., 2012] **Gogouvitis, S., Konstanteli, K., Waldschmidt, S., Kousiouris, G., Katsaros G., Menychtas A., Kyriazis D.,**

Varvarigou, T., Workflow management for soft Real-time Interactive applications in virtualized environments. Future Generation Computer Systems 28 (1), 2012, pp.193–209.

[González and Dasgupta 2003] **Gonzalez, F. A. and Dasgupta, D**. 2003. Anomaly detection using real-valued negative selection. Genetic Programming and Evolvable Machines 4, 4, 383–404.

[González and Ruggia, 2010] **Laura González and Raúl Ruggia.** 2010. Towards dynamic adaptation within an ESB-based service infrastructure layer. In Proceedings of the 3rd International Workshop on Monitoring, Adaptation and Beyond (MONA '10). ACM, New York, NY, USA, pp.40-47.

[Gonzalez and Dasgupta 2003] **Gonzalez, F. A. and Dasgupta, D.** 2003. Anomaly detection using real-valued negative selection. Genetic Programming and Evolvable Machines Vol.4, No.4, 383–404.

[Goldman, 1994] **Goldman, S. L., Nagel, R. N. and Preiss, K., 1994**, Agile Competitors and Virtual Organizations: Strategies for Enriching the Customer (New York: Van Nostrand Reinhold).

[Goepp, 2008] **Virginie Goepp, François Kiefer, Oscar Avila,** Information system design and integrated enterprise modelling through a key-problem framework, Computers in Industry, Volume 59, Issue 7, September 2008, Pages 660-671

[Gunasekaran, 2004] **A Gunasekaran, C Patel, Ronald E McGaughey**, A framework for supply chain performance measurement, International Journal of Production Economics, Volume 87, Issue 3, 18 February 2004, Pages 333-347

[Glinz, 2007] **Glinz, M.,** "On Non-Functional Requirements," Requirements Engineering Conference, 2007. RE '07. 15th IEEE International, pp.21,26, 15-19 Oct. 2007

[Giannis, et al, 2008] **Giannis T. Tsoulfas, Costas P. Pappis**, A model for supply chains environmental performance analysis and decision making, Journal of Cleaner Production, Volume 16, Issue 15, October 2008, Pages 1647-1657

[Haag 2002] **Haag, S., Cummings, M., & McCubbrey, D. J.** (2002). Book: Management information systems for the information age (3rd ed.). McGraw-Hill Companies, Inc.p558.

[Hahn, et al., 1999] **Hahn, G.J., Hill, W.J., Hoerl, R.W., Zinkgraf, S.A.**, 1999. The impact of Six Sigma Improvement—a glimpse into the future of statistics. The American Statistician, Vol.53, No. 3, 208–215.

[Hammer and Champy, 1993] **Michael Hammer, James Champy**, Reengineering the corporation: a manifesto for business revolution, Harper Business, 1993, p223.

[Hämäläinen and Liimatainen, 2008] **Hämäläinen N., Liimatainen K.**, A Framework to Support Business-IT Alignment in Enterprise Architecture Decision Making, in the proceedings of the EBRF conference, 2008, p13.

[Hamer 2002] **Harmer P K, Williams P D, Gunsch G H, Lamont G B,** "An artificial immune system architecture for computer security applications". IEEE Transactions on Evolutionary Computation, Vol. 6, No. 3, pp.252–280, 2002.

[Harry et al., 2011] **Harry H. M. Hendrickx, S. Kevin Daley, Mieke Mahakena, and Mark von Rosing**. 2011. Defining the Business Architecture Profession. In Proceedings of the 2011 IEEE 13th Conference on Commerce and Enterprise Computing (CEC '11). IEEE Computer Society, Washington, DC, USA, 325-332.

[Hirschheim and Sabherwal, 2001] **R. Hirschheim, R. Sabherwal**, Detours in the path toward strategic information systems alignment, California Manage-ment Review, 44 (1) (2001), pp.87–108

[Hilty 2009] **Hilty J.,** "Build the Most Competitieve Enterprise: Leverage the powerful Partnership of BPM and SOA", SentientPoint, Inc.2009

[Huang 2010] **N., Huang, R., Inman**, Product quality and plant build complexity, International Journal of Production Research, vol. 48, Issue.11, pp 3105-3128, 2010

[Heward, 2010] **Heward, G.,** Assessing the Performance Impact of Service Monitoring. In Proceedings of 21st Australian Software Engineering Conference (ASWEC '10). IEEE Computer Society, 2010, pp. 192-201, Washington, DC, USA.

[Hicks 2007] **B.J. Hicks**, Lean information management: Understanding and eliminating waste, International Journal of Information Management, Volume 27, Issue 4, August 2007, Pages 233-249,

[Hammer 1990] **Hammer M..,** Re-engineering work: Don't automate, obliterate. Harvard Business Review. pp.104-112, 1990

[Imran 2008] **Imran S., Rafaqut K., Shahzad M., Shahid N., M. Abbas C.,** [2008] "SOA and BPM Partnership: A paradigm for Dynamic and Flexible Process and I.T. Management", Proc. of World Academy of Science, Engineering and Technology Volume 35 Issue 4 Nov. 2008, pp 16-22

[ITGI 2007] **ITGI**, IT Governance Institute: CObIT 4.1 Excerpt, Executive Summary. 2007.

[IBM 2011] **IBM:** Cloud-enabled business model accelerator. IBM Global Business Services Datasheet. 2011, http://www-935.ibm.com/services/us/gbs/bus/html/cloud.html

[IFIP, 199] **IFIP–IFAC** Task Force on Architectures for Enterprise Integration Generalised Enterprise Reference Architecture and Methodology, Version 1.6.3, 1999. http://www.ict.griffith.edu.au/~bernus/taskforce/geram/versions/geram1-6-3/GERAMv1.6.3.pdf.

[Jayasinghe et al. 2012] **Jayasinghe D., Swint G., Malkowski S.,Li J., Wang Q, Park J., Pu C.,** Expertus: A Generator Approach to Auto-mate Performance Testing in IaaS Clouds, IEEE Fifth International Conference on Cloud Computing, 2012, pp. 115-122

[Jensen et al. 2011] **Claus T. Jensen, Owen Cline, Martin Owen**, Combining Business Process Management and Enterprise Architecture for Better Business Outcomes, IBM Redbooks, 2011

[Johnson and Swisher, 2003] **Johnson, A., Swisher, B.**, 2003. How six sigma improves R&D. Research Technology Management 46 (2), 12-15.

[J2CA 2003] **J2CA Group.**: J2EE Connector Architecture Specification, version 1.5. Technical report, SUN Microsystems, 2003

[Jasmine 2005] **Jasmine N.**, "BPM and SOA: Better Together", IBM White Paper, pp. 1-12, 2005.

[Joachim et al., 2013] **Joachim N., Beimborn D., Weitzel T.**, The influence of SOA governance mechanisms on IT flexibility and service reuse, In Journal of Strategic Information Systems, Vol. 22, No. 1, pp86-101, 2013

[Jung et al., 2004] **H.W. Jung, S.G. Kim, C.-S. Chung**, Measuring software product quality: a survey of ISO7IEC 9126, IEEE Software 21 (2004) 88–92.

[Kamoun 2007] **Kamoun F.**, The Convergence of Business Process Management and Service Oriented Architecture, Ubiquity archive, Article No.3 ACM New York, USA, 2007

[Katsaros et al., 2012] **Katsaros, G., Kousiouris, G., Gogouvitis, S.V., Kyriazis D., Menychtas, A., Varvarigou T.,** 2012. A Self adaptive hierarchical monitoring mechanism for Clouds, Journal of Systems and Software, 85 (5), 2012., pp.1029-1041,

[Kaplan and Norton, 2001] **Kaplan R., Norton D.**, Book: The strategy-focused organization, Havard Business School Press, Massachusetts,2001. 400p.

[Karlsson et al. 2007] **Lena Karlsson, Åsa G. Dahlstedt, Björn Regnell, Johan Natt och Dag, Anne Persson**, Requirements engineering challenges in market-driven software development – An interview study with practitioners, Information and Software Technology, Volume 49, Issue 6, June 2007, Pages 588-604

[Ketchen, 2008] **David J. Ketchen, Jr., William Rebarick, G. Tomas M. Hult, David Meyer,** Best value supply chains: A key competitive

weapon for the 21st century, Business Horizons, Volume 51, Issue 3, May–June 2008, Pages 235-243,

[KEKE, 2012] **KEKE** WP5, Hasse Nylund, Kaushik Shankar, Mikko Koho, Discussion on Lean, Agile and Sustainable, 24.4.2012

[Kim 2009] **Kim W.**, Cloud Computing: Today and Tomorrow, In Journal of Object Technology, Vol.8, No. 1, pp. 65-72, 2009

[Kidd, 1994] **Kidd, T.,** (1994). "Agile Manufacturing: Forging new frontiers". Addison-Wesley Reading, MA.

[Kooper 2011] **M. N. Kooper, R. Maes, and E.E.O. Roos Lindgreen**. 2011. On the governance of information: Introducing a new concept of governance to support the management of information. Int. J. Inf. Manag. 31, 3 (June 2011), 195-200.

[Ko, 2009] **Ryan K. L. Ko.** 2009. A computer scientist's introductory guide to business process management (BPM). Crossroads 15, 4, Article 4 (June 2009), 8 pages.

[Koho, 2010] **Koho, M., Nylund, H., Arha, T. and Torvinen, S.** 2010. "Towards Manufacturing System Sustainability Assessment: An Initial Tool and Development Plans", 8th Global Conference on Sustainable Manufacturing, Abu Dhabi, UAE, 22 - 24 November, 309-314.

[Kraaijenbrink 2007] **Kraaijenbrink, J.** (2007). Engineers and the web: An analysis of real life gaps in information usage. Information Processing and Management, 43(5), 1368–1382.

[Kwak and Anbari, 2006] **Young Hoon Kwak, Frank T. Anbari**, Benefits, obstacles, and future of six sigma approach, Technovation, Volume 26, Issues 5–6, May–June 2006, Pages 708-715,

[Ken and Herry, 2006]**Ken V. , Henry P.**, "The Forrester Wave: Integration-Centric Busi-ness Process Management Suites", Forrester Research Inc. Re-port, Q4, pp. 1-16, 2006

[Kennerley and Neely, 2003] **Mike Kennerley, Andy Neely**, (2003) "Measuring performance in a changing business environment",

International Journal of Operations & Production Management, Vol. 23 Iss: 2, pp.213 – 229

[Lankhorst M, et al. 2005] **Lankhorst M, et al.** Enterprise Architecture at Work: Modelling, Communication and Analysis, Springer 2005. p334

[Lee and Choi, 2006] **Kun-chang Lee, Bong Choi**, Six Sigma management activities and their influence on corporate competitiveness, Total Quality Management, Vol. 17, No.7, 2006, pp.893-911.

[Leganza 2004] **Leganza, G.**, Top-Down Versus Bottom-Up: Approaches to Enterprise Architecture, 2004, http://www.forrester.com/research/document/excerpt/0,7211,34080,00.html, accessed in 2004

[Lohman et al. 2004] **Lohman C., Fortuin L., Wouters M.**, Designing a performance measurement system: a case study, European Journal of Operational Research 156 (2004) 267–286.

[Loucopoulos et al. 2013] **Loucopoulos P, Sun J, Zhao L, Heidari F.** A systematic classification and analysis of NFRs 19th Americas conference on information systems (AMCIS 2013). Chicago: USA AIS; 2013

[CIMOSA, 1993] **CIMOSA**: Open System Architecture for CIM, AMICE Consortium, ESPRIT Research Reports, Volume 1, 1993, Springer, ISBN: 978-3-540-56256-6

[Liker 2004] **J.K. Liker,** Book: The Toyota way: 14 management principles from the world's greatest manufacturer, published by McGraw-Hill, New York (2004), 350p.

[Luis M., 2001] **Luis M. Sanchez & Rakesh Nagi** (2001) A review of agile manufacturing systems, International Journal of Production Research, 39:16,3561-3600

[Marchand et al. 2000] **Marchand, D., Davenport, T., and Dickson, T.** (2000). Book: Mastering information management, published by financial times. London: Prentice Hall. p.362

[Marchand and Raymond 2008] **Marchand, M., and Raymond, L.** (2008). Researching performance measurement systems – An information systems perspective. International Journal of Operations and Production Management, 28(7), 663–686.

[Malveau 2004] **Malveau, R.**, Bridging the Gap: Business and Software Architecture, Part 2, 2004, Cutter Consortium, www.cutter.com/research/2004/edge040203.html, accessed in 2004

[Marks and Bell 2006] **Marks E., Bell M.,** Book: "Service-Oriented Architecture: A Planning and Implementation Guide for Business and Technology", published by John Wiley & Sons, Inc., New Jersey, 2006. 400p.

[Marston et al. 2011] **Sean Marston, Zhi Li, Subhajyoti Bandyopadhyay, Juheng Zhang, and Anand Ghalsasi.** 2011. Cloud computing - The business perspective. Decis. Support Syst. 51, 1 (April 2011), 176-189.

[Mason-Jones, 2000] **Mason-Jones, R., Naylor, B. & Towill, D.R.** 2000. Lean, agile or leagile? Matching your supply chain to the marketplace. International Journal of Production Research 38, 17, pp.4061-4070.

[McCormack and Johnson, 2010] **McCormack, K., Johnson W.**, Book: Business Process Orientation, published by CRC Press, 208p. 2010

[McFadden 2012] **McFadden P.,** CEO of ExcelDashboardWidgets "What is Dashboard Reporting", 2012.05

[Melville et al. 2004] **N. Melville, K. Kraemer, V. Gurbaxani**, Information technology and organizational performance: An integrative model of IT business value MIS Quarterly, 28 (2) (2004), pp. 283–322

[Mell 2011] **Mell P, Grance T.** The NIST Definition of Cloud Computing. Recommendations of the National Institute of Standards and Technology, NIST Special Publication 800-145, January 2011

[Medori and Steeple, 2000] **Medori D., Steeple D.** , A framework for auditing and enhancing performance measurement systems, International Journal of Operations and Production Management 20 (5) (2000) 520–533

[Morganwalp and Sage, 2004] **Morganwalp, J. M., Sage, A. P.,** Enterprise Architecture Measures of Effectiveness. International Journal of Technology, Policy and Management, Vol. 4, No. 1, pp. 81-94, 2004.

[Myerson 2012] **Paul Myerson**, Lean supply chain and logistics management, McGrawHill, 2012, p292.

[Millard 2001] **Millard, R. L**. Value Stream Analysis and Mapping for ProductDevelopment, Cambridge, MA: Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, master thesis (2001)

[Mos et al. 2008] **Mos, A., Boulze, A., Quaireau, S., and Meynier, C.**,. 2008. Multi-layer perspectives and spaces in SOA. In Proceedings of the 2nd international workshop on Systems development in SOA environments (SDSOA '08). ACM, New York, NY, USA, 69-74.

[OMG 2003] **OMG**. (2003, 06 12). MDA Guide Version 1.0.1. Retrieved from Object Management Group: http://www.omg.org/mda

[Nagel and Dove, 1991] **Roger N. Nagel, Rick Dove,** 21st Century Manufacturing Enterprise Strategy: An Industry-Led View, DIANE Publishing, 1991, p58

[Neely et al. 1995] **Neely, A., Mills, J., Gregory, M., & Platts, K.** (1995). Performance measurement system design – A literature review and research agenda. International Journal of Operations and Production Management, 15(4), 80–116.

[Neely et al. 2000] **Neely A., Mills J., Platts K., Richards H., Gregory M., Bourne M., Kennerley M.**, Performance measurement system design: developing and testing a process-based approach, International Journal of Operations and Production Management 20 (10) (2000) 1119–1145.

[Negash and Gray 2008] **Negash S, Gray P.,** Business intelligence, international handbook on information systems. Handbook on decision support systems. Berlin Heidelberg: Springer; 2008. p.175—193.

[Niemann et al. 2008] **Niemann M., Eckert J., Repp N., Steinmetz R.**, "Towards a Generic Governance Model for Service Oriented Architectures",AMCIS 2008 Proceedings, Paper 361. 2008,http://aisel.aisnet.org/amcis2008/361

[Niemann et al. 2009] **Michael Niemann, Christian Janiesch, Nicolas Repp, and Ralf Steinmetz.** Challenges of Governance Approaches for Service-Oriented Architectures. In Proceedings of the Third International Conference on Digital Ecosystems and Technologies (DEST 2009), pages 634–639, Istanbul, Turkey, June 2009.

[Nudurupati et al. 2011] **Nudurupati, S.S., Bititci, U.S., Kumar, V., Chan, F.T.S.** : State of the art literature review on performance measurement, J. Computers & Industrial Engineering, Vol. 60, No. 2, pp. 279-290.(2011)

[OASIS 2008] **OASIS**, Reference Architecture for Service Oriented Architecture Version 0.3, March 4, 2008, p102

[OASIS RM 2006] **OASIS** Reference Model for Service Oriented Architecture 1.0, Official OASIS Standard, Oct. 12, 2006

[Pauwels, et al. 2009] **Pauwels, K., Ambler, T., Bruce, H.C., Lapointe, P., Reibstein, D., Skiera, B., Wierenga, B., Wiesel, T.**, Dashboards as a Service: Why, How, and What Research Is Needed ? , Journal of Service Research, Vol. 12, No.2, pp. 175-189, 2009

[Pamfilie 2012] **Rodica Pamfilie, Andreea Jenica Petcu (Draghici), Mihai Draghici**, The Importance of Leadership in Driving a Strategic Lean Six Sigma Management, Procedia - Social and Behavioral Sciences, Volume 58, 12 October 2012, Pages 187-196,

[Papazoglou et al. 2006] **Papazoglou M., Van Den W., Heuvel**, Service-oriented design and development methodology. Int. J. Web Eng. Technol. Vol.2, No.4, 2006, pp.412-442.

[Papaoglou and Heuvel, 2007] **Papazoglou, MP., Heuvel, WJ,**. Serivce oriented architec-tures: approaches, technologies and research issues, The Inter-national Journal on Very Large Data Bases, Vol. 16, No.3, pp.389-415, Springer-Verlag NY, USA, July 2007

[Pavlou and EL Sawy 2006] **Pavlou P., EI Sawy, O.,** From IT leveraging competence to com-petitive advantage in turbulent environments: the case of new product development. Information Systems Research, In In-formation System Research, Vol. 17, No. 3, pp. 198-227, 2006

[Pamfilie 2012] **Rodica Pamfilie, Andreea Jenica Petcu (Draghici), Mihai Draghici,** The Importance of Leadership in Driving a Strategic Lean Six Sigma Management, Procedia - Social and Behavioral Sciences, Volume 58, 12 October 2012, Pages 187-196,

[Perks and Beveridge, 2003] **Col Perks, Tony Beveridge**, Guide to Enterprise IT Architecture, Springer 2003, p447

[Porter, 2008] **Michael E. Porter,** Competitive Advantage: Creating an Sustaining Superior Performance, Simon and Schuster-Business & Economics, 2008, p592.

[Powell et al., 2013] **Daryl Powell, Erlend Alfnes, Jan Ola Strandhagen, Heidi Dreyer**, The concurrent application of lean production and ERP: Towards an ERP-based lean implementation process, Computers in Industry, Volume 64, Issue 3, April 2013, Pages 324-335,

[Pyzdek and Keller, 2009] **Pyzdek, T., Keller, P.**, 2009. Book: The Six Sigma Handbook: A Complete Guide for Green Belts, Black Belts, and Managers at All Levels, 3rd ed. McGraw Hill, New York,NY.p.711

[Qrunfleh, 2014] **Sufian Qrunfleh, Monideepa Tarafdar,** Supply chain information systems strategy: Impacts on supply chain performance and firm performance, International Journal of

Production Economics, Volume 147, Part B, January 2014, Pages 340-350,

[Raschke 2010] **Robyn L. Raschke**, Process-based view of agility: The value contribution of IT and the effects on process outcomes, International Journal of Accounting Information Systems, Volume 11, Issue 4, pp. 297-313, Decembe 2010

[Raschke and David, 2005] **Raschke, Robyn L. and David, Julie Smith**, "Business Process Agility" (2005). AMCIS 2005 Proceedings. Paper 180.

[Rahman and Ripon, 2013] **Md. Mijanur Rahman and Shamim Ripon**, "Elicitation and Modeling Non-Functional Requirements – A POS Case Study," International Journal of Future Computer and Communication vol. 2, no. 5, pp. 485-489, 2013.

[Rosing et al. 2011] **Mark von Rosing, Raghavendra 'Rao' Subbarao, Maria Hove, Tom W. Preston,** Combining BPM and EA in complex ERP projects, In proceeding of: 13th IEEE Conference on Commerce and Enterprise Computing, CEC 2011, Luxembourg-Kirchberg, Luxembourg, September 5-7, 2011

[Saiz et al. 2005] **Saiz, J.J.A., Rodriguez, R.R. Bas, A.O.,** : A performance measurement system for virtual and extended enterprises. In: IFIP International Federation for Information Processing, pp. 285- 292. (2005)

[Sambamufthy et al. 2003] **Sambamurthy, V., Bharadwaj, A., and Grover, V.** "Shaping Agility through Digital Options: Reconceptualizing the Role of Information Technology in Contemporary Firms," MIS Quarterly (27:2), 2003, pp. 237-263.

[Scherrer-Rathje and Boyle, 2009] **Maike Scherrer-Rathje, Todd A. Boyle**, Patricia Deflorin, Lean, take two! Reflections from the second attempt at lean implementation, Business Horizons, Volume 52, Issue 1, 2009, pp.79-88

[Schonberger 2005] **R. Schonberger**, Lean extended: It's much more (and less) than you think Industrial Engineer, 37 (12) (2005), pp. 26–31

[Schroeder et al., 2008] **Roger G. Schroeder, Kevin Linderman, Charles Liedtke, Adrian S. Choo,** Six Sigma: Definition and underlying theory, Journal of Operations Management, Volume 26, Issue 4, July 2008, Pages 536-554,

[Schepers et al. 2008] **Schepers, T.G.J., Iacob, M.E., van Eck, P.A.T.**, 2008. A lifecycle approach to SOA governance. In: Proceedings of the 2008 ACM Symposium on Applied computing. Fortaleza, Ceara, Brazil: ACM, pp. 1055–1061.

[Seethamraju and Sundar 2013] **Ravi Seethamraju, Diatha Krishna Sundar**, Influence of ERP systems on business process agility, IIMB Management Review, pp.1-13

[Sengupta and Masini 2008] **Kishore Sengupta, Andrea Masini,** It Agility: Striking the Right Balance. Business Strategy Review, Vol. 19, Issue 2, pp. 42-48, Summer 2008.

[Sessions 2007] **Sessions R**, comparison of the top four enterprise architecture methodologies, May 2007, http://msdn.microsoft.com/en-us/library/bb466232.aspx

[Sharifi and Zhang 2001] **H. Sharifi, Z. Zhang,** "Agile manufacturing in practice - Application of a methodology", International Journal of Operations & Production Management, Vol. 21 Iss: 5/6, 2001, pp.772 – 794

[Swink and Jacobs, 2012] **Morgan Swink, Brian W. Jacobs**, Six Sigma adoption: Operating performance impacts and contextual drivers of success, Journal of Operations Management, Volume 30, Issue 6, September 2012, Pages 437-453

[Škrinjar and Trkman 2013] **Škrinjar, R., Trkman, P.**, Increasing process orientation with business process management: Critical practices', International Journal of Information Management, Volume 33, Issue 1, February 2013, Pages 48-60,

[SLA 2006] **SLA** Management Handbook, Vol.4, Enterprise Perspective, Telemanagement forum, the Open Group, 2006

[Sliman 2008] **Sliman L**., Thesis: C-Business et urbanization d'enterprise, 2008 http://theses.insa-lyon.fr/publication/2009ISAL0099/these.pdf

[Sterrit, 2005] **Sterritt R.,** Automonic Computing, Journal of Innovations in Systemsand Software Engineering, Vol.1, No. 1, pp. 79-88, 2005

[Sultan 2013] **Sultan, N.,** Cloud computing: A democratizing force?, International Journal of Information Management, Volume 33, Issue 5, October 2013, Pages 810-815,

[Svensson et al. 2013] **Richard Berntsson Svensson, Thomas Olsson, Björn Regnell**, An investigation of how quality requirements are specified in industrial practice, Information and Software Technology, Volume 55, Issue 7, July 2013, Pages 1224-1236,

[Svensson, et al. 2012] **R. Berntsson Svensson, T. Gorschek, B. Regnell, R. Torkar, A. Shahrokni, R. Feldt**, Quality requirements in industrial practice – an extended interview study at eleven companies, IEEE Transaction on Software Engineering 38 (2012) 923–935.

[Svensson 2009] **R. Berntsson Svensson, T. Gorschek, B. Regnell**, Quality requirements in practice: an interview study in requirements engineering for embedded systems, in: M. Glinz, P. Heymans, (Eds.), REFSQ 2009, LNCS, vol. 5512, Springer, Heidelberg, pp. 218–232.

[Tallon 2008] **Paul Patrick Tallon**, Inside the adaptive enterprise: an information technology capabilities perspective on business process agility, Information technology and management, Vol. 9, No. 1, 2008, pp. 21-36

[Tarakanov 2008] **Tarakanov A O,** "Immuno computing for intelligent intrusion detection", IEEE Computational Intelligence Magazine, Vol. 3, No. 2, pp(s):22–30, 2008.

[Teece 2010] **Daved J. Teece**, Business Models, Business Strategy and Innovation, Long Range Planning, 43 (2010) pp. 172-194

[TOGAF 2003] **TOGAF** - version 8, 2003, The Open Group, http://www.opengroup.org/architecture/togaf8-doc/arch/,

[TM Forum 2008] **Telemanagement (TM) Forum**. SLA Handbook Solution Suite V2.0. http://www.tmforum.org/DocumentLibrary/SLAHandbookSolution/2035/Home.htmlH (2008).

[Tsai 2013] **Tsai, J.Y., Shao, B.B.M.,** Information systems and technology sourcing strategIES OF e-retailERS FOR value chain Enablement, Journal of Operations Management (2013),

[Tsakonas and Paptheodorou, 2008] **G. Tsakonas, C. Papatheodorou,** "Exploring usefulness and usability in the evaluation of open access digital libraries", Information Processing and Management Journal, Vol. 44(3), pp. 1234-1250, 2008.

[van Oosterhout et al., 2007] **Marcel van Oosterhout, Eric Waarts, Eric van Heck, and Jos van Hillegersberg,** Business Agility: Need, Readiness and Alignment with IT Strategies, In Agile Information Systems: Conceptualization, Construction, and Management, pp. 52-69 Elsevier Inc. 2007

[van der Aalst, et al., 2003] **Wil M. P. Van Der Aalst, Arthur H. M. Ter Hofstede, and Mathias Weske**. Business process management: a survey. In Proceedings of the 2003 international conference on Business process management (BPM'03), Springer-Verlag, Berlin, Heidelberg, 2003. pp.1-12.

[van der Stede, et al., 2006] **Wim A. Van der Stede, Chee W. Chow, and Thomas W. Lin** (2006) Strategy, Choice of Performance Measures, and Performance. Behavioral Research in Accounting: February 2006, Vol. 18, No. 1, pp. 185-205.

[Vasilakos et al. 2008] **Vasilakos, A., Parashar, M., S.Karnouskos, and W.Pedrycz.** 2008. Autonomic Communication. Springer, p374

[Vara, et al. 2011] **J.L. de la Vara, K. Wnuk, R. Berntsson Svensson, J. Sánchez, B. Regnell**, An empirical study on the importance of quality requirements in industry, in: Proceedings of the 23rd

International Conference on Software Engineering and Knowledge, Engineering, SEKE'11, 2011, pp. 438–443.

[Venkatesan et al. 2013] **S. Venkatesan, R. Baskaran, C. Chellappan, Anurika Vaish, P. Dhavachelvan,** Artificial immune system based mobile agent platform protection, Computer Standards & Interfaces, Volume 35, Issue 4, June 2013, Pages 365-373

[Vernadat 2002] **F.B. Vernadat,** Enterprise modeling and integration (EMI): Current status and research perspectives, Annual Reviews in Control, Volume 26, Issue 1, 2002, Pages 15-25,

[Vukšić et al. 2013] **Vukšić, VB., Bach, MP., Popovič, A**., Supporting performance management with business process management and business intelligence: A case analysis of integration and orchestration, International Journal of Information Management, Volume 33, Issue 4, August 2013, Pages 613-619,

[Vlachos, 2014] **Ilias P. Vlachos**, A hierarchical model of the impact of RFID practices on retail supply chain performance, Expert Systems with Applications, Volume 41, Issue 1, January 2014, Pages 5-15

[Weill and Ross 2004] **Weill, P., and Ross, J.**, How Top Performers Manage IT Decision Rights for Superior Results, Havard Business School Press, 2004, p15.

[Wadhwa and Rao 2003] **Wadhwa, S. and Rao, K. S.** (2003). Flexibility and agility for enterprise synchronization: Knowledge and innovation management towards flexagility. Studies in Informatics and Control, 12 (2), 111–128.

[Wiersma 2009] **Wiersma, E**. (2009): "For which purposes do managers use Balanced Scorecards? An empirical study", Management Accounting Research, vol 20: 239-251.

[Williams, 1994] **Williams, T.J.** (1994), The Purdue Enterprise Reference Architecture, Computers in Industry, 24 (2-3), 141-158

[Womack and Jones 1996] **James P. Womack and Daniel T. Jones**, Lean Thinking: banish waste and create wealth in your corporation, Simon & Schuster, 1996, p350

[Womack et al., 1990] **James P. Womack, Daniel T. Jones, and Daniel Roos**, The Machine That Changed the World, Free Press, 1990, p352.

[W. “RP” Raghupathi 2007] **W. "RP" Raghupathi**. 2007. Corporate governance of IT: a framework for development. Commun. ACM 50, 8 (August2007), 94-99.

[WLM 2007] “Web Layout Mining **(WLM)**: A Paradigm for Intelligent Web Layout Design”, Egyptian Computer Science Journal, Vol. 29,No. 2, May 2007

[Xu X 2012] **Xu X.** From cloud computing to cloud manufacturing. Robotics and computer-Integrated manufacturing, Vol. 28.No.1, P.75-86.2012

[Young et al. 2001] **C. Young, Y. Lakshman, T. Szymanski, J. Reppy, D. Presotto, R. Pike, G. Narlikar, S. Mullender, E. Grosse, Protium**, an infrastructure for partitioned applications, in: Hot Topics in Operating Systems, 2001. Proceedings of the Eighth Workshop on,IEEE, 2001, pp. 47–52.

[Yusuf et al., 2004] **Yusuf, Y. Y., et al.** (2004). Agile supply chain capabilities: Determinants of competitive objectives. European Journal of Operational Research, 159 (2), 379–392.

[Yigitbasioglu and Velcu 2012a] **Yigitbasioglu, O.M., Velcu, O.,** A review of dashboards in performance management: Implications for design and research, International Journal of Accounting Information Systems, Volume 13, Issue 1, March 2012, Pages 41-59,

[Yigitbasioglu and Velcu 2012b] **Yigitbasioglu, O.M., Velcu, O.,** The use of dashboards in performance management: evidence from sales managers. The International Journal of Digital Accounting Research, 12, pp. 39-58. 2012

[Zachman 1987] **Zachman J.**, A framework for information systems architecture, JOURNAL ARTICLE, IBM Systems Journal, pp. 276-292,1987.

[Zachman and Sowa 1992] **Zachman, J., Sowa J.F.**, Extending and Formalizing the Framework for Information Systems Architecture. IBM Systems Journal, 31(3), pp. 590-616, 1992.

[Zachman 2003] **John A. Zachman**, The Zachman Framework for enterprise architecture: Primer for Enterprise Engineering and Manufacturing, 2003. http://www.zachmaninternational.com

[Zhang et al., 2010] **Qi Zhang, Lu Cheng, Raouf Boutaba**, Cloud Computing: state-of-the-art and research challenges, Journal of Internet Services and Applications, Vol. 1, No. 1, 2010. pp.7-18

[Zowghi, Coulin, 2005] **D. Zowghi, C., Coulin**, Requirements Elicitation: A survey of Techniques, Approaches, and Tools, Engineering and Managing Software Requirements, 2005

Appendix

1 Use Case - BaaS Management: Definition and Dependencies of Resources for Standard Delivery BP

The formalized definition of Standard Delivery BP's resources as following:

BP standard = ("Standard Delivery BP", "Standard: Deliver goods from factory to clients", "Task 1, Task 2, Task 3, Task 4", "Delivery time: Standard, Cost: Low", "BP_Cxt", "BP_Ep")

This BP is divided into four Tasks accordingly:

Task 1 = ("Task 1", "distribution from factory to depot", "Service 1", "Delivery time, Cost", "Task_Cxt", "Task_Ep");

Task 2 = ("Task 2", "distribution from depot to distribution center", "Express: Service E2; Standard: S2", "Delivery time, Cost", "Task_Cxt", "Task_Ep");

Task 3 = ("Task 3", "distribution from distribution center to transmit point", "Express: Service E3; Standard: S3", "Delivery time, Cost", "Task_Cxt", "Task_Ep");

Task 4 = ("Task 4", "distribution from transmit point to client", "Express: Service E4; Standard: S4", "Delivery time, Cost", "Task_Cxt", "Task_Ep");

Each Task requires at least one Service, in this case there are four Services are required:

Service 1 = ("Service 1", "distribution from factory to depot", "black-box", "FP: shipping", "NFP: delivery time; cost", "", "Service1_Ep")

Service S2 = ("Service S2", "distribution from depot to distribution center", "white-box", "FP: shipping; Require: OP S2-1; OP S2-2", "NFP: delivery time; cost", "", "ServiceS2_Ep")

Service S3 = ("Service S3", "distribution from distribution center to transmit point", "white-box", "FP: shipping; Require: OP S3-1; OP S3-2", "NFP: delivery time; cost", "", "ServiceS3_Ep")

Service S4 = ("Service S4", "distribution from transmit point to client", "white-box", "FP: shipping; Require: OP S4-1; OP S4-2", "NFP: delivery time; cost", "", "ServiceE4_Ep")

Each Service is implemented by at least on Operation, in this case there are four Operations are required:

Operation S2-1 = ("OP S2-1", "manage inventory", "FP: inventory management", "NFP", "", "OP S3-1_Ep");

Operation S2-2 = (“OP S2-2”, “manage shipment”, “FP: shipment management”, “NFP”, “”, “OP S3-2_Ep”);

Operation S3-1 = (“OP S3-1”, “manage inventory”, “FP: inventory management”, “NFP”, “”, “OP S3-1_Ep”);

Operation S3-2 = (“OP S3-2”, “manage shipment”, “FP: shipment management”, “NFP”, “”, “OP S3-2_Ep”);

Operation S4-1 = (“OP S4-1”, “manage inventory”, “FP: inventory management”, “NFP”, “”, “OP S4-1_Ep”);

Operation S4-2 = (“OP S4-2”, “manage shippment”, “FP: shippment management (Require: Inf S4-2)”, “NFP”, “”, “OPS4-2_Ep”);

For white-box Operation, the required Infrastructure can be managed. In this case there are three Infrastructures can be managed.

Infrastructure S2-2 = (“Inf S2-2”, “implement OP S2-2”, “shipping”)

Infrastructure S3-2 = (“Inf S3-2”, “implement OP S3-2”, “shipping”)

Infrastructure S4-2 = (“Inf S4-2”, “implement OP S4-2”, “shipping”)

From above, the dependencies of these Standard Delivery BP’s resources are shown in following figure (see Figure 88):

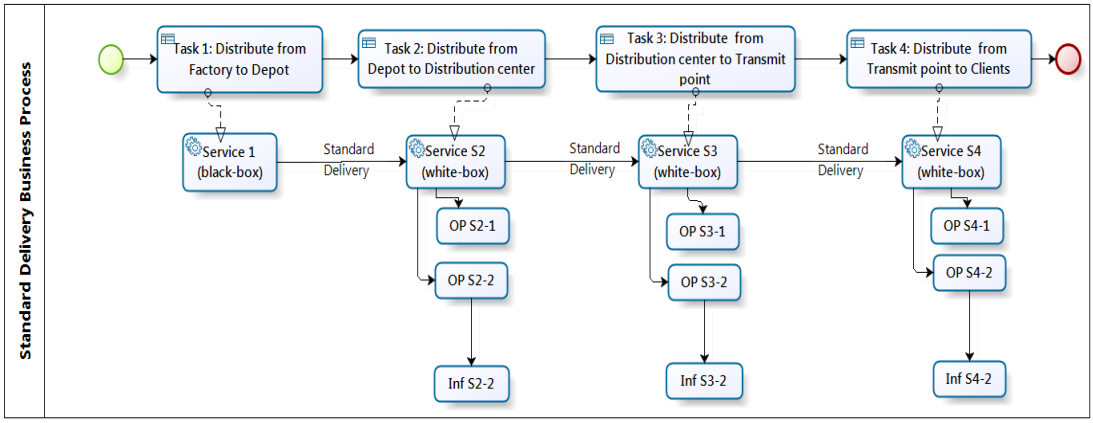


Figure 88 *Standard Delivery BP's Resource Dependency*

2 Use Case (Governance Preparation) – Classification of NFP, Transform Pattern and Governance Pattern

In our Use Case Non Functional Properties are classified into four groups, each groups has at least one Critical Success Factors (see Figure 89):

Figure 89 *Excerpt of NFP's Classification xml file*

Details of sub-group Delay Rate (see Figure 90):

```
<NFP>
  <name>subGroup-DelayRate</name>
  <Elements>
    <Element>number of delayed time</Element>
    <Element>number of measured time</Element>
  </Elements>
  <Measurement>delayrate.value=( number of delayed time/ number of measured time )*100%</Measurement>
  <Algorithm>
  <ParenetNFP>
    <name>Group-Performance</name>
  </ParenetNFP>
  <ChildNFP>CSF-ResponseDelayRate</ChildNFP>
  <ChildNFP>CSF-ExecutionDelayRate</ChildNFP>
</NFP>
```

Sub Group – Delay Rate

```
<NFP>
  <name>CSF-ResponseDelayRate</name>
  <Elements>
    <Element>number of delayed time</Element>
    <Element>number of measured time</Element>
  </Elements>
  <Measurement>delayrate.value=( number of delayed time/ number of measured time )*100%</Measurement>
  <Algorithm>
  <ParenetNFP>subGroup-DelayRate</ParenetNFP>
</NFP>
<NFP>
  <name>CSF-ExecutionDelayRate</name>
  <Elements>
    <Element>number of delayed time</Element>
    <Element>number of measured time</Element>
  </Elements>
  <Measurement>delayrate.value=( number of delayed time/ number of measured time )*100%</Measurement>
  <Algorithm>
  <ParenetNFP>subGroup-DelayRate</ParenetNFP>
</NFP>
```

CSF – Response Delay Rate

CSF – Execution Delay Rate

Figure 90 *Details of sub-group Delay Rate*

Transformation patterns are organized based on the Non Functional Property classification (see Figure 91, Figure 92, Figure 93, Figure 94):

Pattern-Cost

```
<TransformPattern>
  <PatternName>Pattern-Cost</PatternName>
  <PatternGoal>Govern-cost</PatternGoal>
  <PatternContext>pattern context--</PatternContext>
  <PatternPariticipants>paticipants:Req i; subpatterns; collaborated BP; etc.</PatternPariticipants>
  <PatternCollaboration>collaboration of participants</PatternCollaboration>
  <Relatedpatterns>
    <relatedSiblingpatterns>
      <siblingpattern>Pattern-Performance</siblingpattern>
      <siblingpattern>Pattern-Maintainability</siblingpattern>
      <siblingpattern>Pattern-Security</siblingpattern>
    </relatedSiblingpatterns>
    <relatedChildpatterns>
      <childpattern>CSFPattern-Price</childpattern>
    </relatedChildpatterns>
  </Relatedpatterns>
  <PatternConsequences>
  <PatternStep>transform steps</PatternStep>
</TransformPattern>
```

Figure 91 *Transform Pattern - Cost*

Pattern-Performance

```
<TransformPattern>
  <PatternName>Pattern-Performance</PatternName>
  <PatternGoal>Govern-performance</PatternGoal>
  <PatternContext>pattern context--</PatternContext>
  <PatternPariticipants>paticipants:Req i; subpatterns; collaborated BP; etc.</PatternPariticipants>
  <PatternCollaboration>collaboration of participants</PatternCollaboration>
  <Relatedpatterns>
    <relatedSiblingpatterns>
      <siblingpattern>Pattern-Cost</siblingpattern>
      <siblingpattern>Pattern-Maintainability</siblingpattern>
      <siblingpattern>Pattern-Security</siblingpattern>
    </relatedSiblingpatterns>
    <relatedChildpatterns>
      <childpattern>CSFPattern-ResponseTime</childpattern>
      <childpattern>CSFPattern-Availability</childpattern>
      <childpattern>subPattern-DelayRate</childpattern>
      <childpattern>CSFPattern-ExecutionTime</childpattern>
    </relatedChildpatterns>
  </Relatedpatterns>
  <PatternConsequences>
</TransformPattern>
```

Figure 92 *Transform Pattern - Performance*

Pattern-Maintainability

```
<TransformPattern>
  <PatternName>Pattern-Maintainability</PatternName>
  <PatternGoal>Govern-maitainability</PatternGoal>
  <PatternContext>pattern context--</PatternContext>
  <PatternPariticipants>paticipants:Req i; subpatterns; collaborated BP; etc.</PatternPariticipants>
  <PatternCollaboration>collaboration of participants</PatternCollaboration>
  <Relatedpatterns>
    <relatedSiblingpatterns>
      <siblingpattern>Pattern-Cost</siblingpattern>
      <siblingpattern>Pattern-Performance</siblingpattern>
      <siblingpattern>Pattern-Security</siblingpattern>
    </relatedSiblingpatterns>
    <relatedChildpatterns>
      <childpattern>CSFPattern-Reputation</childpattern>
      <childpattern>CSFPattern-Usability</childpattern>
      <childpattern>CSFPattern-Reliability</childpattern>
      <childpattern>CSFPattern-Accuracy</childpattern>
    </relatedChildpatterns>
  </Relatedpatterns>
  <PatternConsequences>
</TransformPattern>
```

Figure 93 *Transform Pattern - Maintainability*

Pattern-Security

```
<TransformPattern>
  <PatternName>Pattern-Security</PatternName>
  <PatternGoal>Govern-security</PatternGoal>
  <PatternContext>pattern context--</PatternContext>
  <PatternPariticipants>paticipants:Req i; subpatterns; collaborated BP; etc.</PatternPariticipants>
  <PatternCollaboration>collaboration of participants</PatternCollaboration>
  <Relatedpatterns>
    <relatedSiblingpatterns>
      <siblingpattern>Pattern-Cost</siblingpattern>
      <siblingpattern>Pattern-Performance</siblingpattern>
      <siblingpattern>Pattern-Maintainability</siblingpattern>
    </relatedSiblingpatterns>
    <relatedChildpatterns>
      <childpattern>subPattern-Integrity</childpattern>
      <childpattern>subPattern-Confidentiality(AES)</childpattern>
      <childpattern>subPattern-Confidentiality(DES)</childpattern>
      <childpattern>subPattern-Nonrepuliation</childpattern>
    </relatedChildpatterns>
  </Relatedpatterns>
  <PatternConsequences>
</TransformPattern>
```

Figure 94 *Transform Pattern - Security*

For subgroup DelayRate, the corresponding subPattern-DelayRate and its CSF-Patterns are organized as following shows (see Figure 95):

subPattern-DelayRate

```
<TransformPattern>
  <PatternName>subPattern-DelayRate</PatternName>
  <PatternGoal>Govern-delay-rate</PatternGoal>
  <PatternContext>pattern context--</PatternContext>
  <PatternPariticipants>paticipants:Req i; subpatterns; collaborated BP; etc.</PatternPariticipants>
  <PatternCollaboration>collaboration of participants</PatternCollaboration>
  <Relatedpatterns>
    <relatedParentpatterns>
      <parentpattern> pattern-performance</parentpattern>
    </relatedParentpatterns>
    <relatedSiblingpatterns>
      <siblingpattern>CSFPattern-ResponseTime</siblingpattern>
      <siblingpattern>CSFPattern-ExecutionTime</siblingpattern>
      <siblingpattern>CSFPattern-Availability</siblingpattern>
    </relatedSiblingpatterns>
    <relatedChildpatterns>
      <childpattern>CSFPattern-ResponseDelayRate</childpattern>
      <childpattern>CSFPattern-ExecutionDelayRate</childpattern>
    </relatedChildpatterns>
  </Relatedpatterns>
  <PatternConsequences>
</TransformPattern>
```

CSFPattern-ResponseDelayRate

```
<TransformPattern>
  <PatternName>CSFPattern-ResponseDelayRate</PatternName>
  <PatternGoal>Govern-delay-rate</PatternGoal>
  <PatternContext>pattern context--</PatternContext>
  <PatternPariticipants>paticipants:Req i; subpatterns; collaborated BP; etc.</PatternPariticipants>
  <PatternCollaboration>collaboration of participants</PatternCollaboration>
  <Relatedpatterns>
    <relatedParentpatterns>
      <parentpattern> Pattern-Performance</parentpattern>
    </relatedParentpatterns>
    <relatedSiblingpatterns>
      <siblingpattern>CSFPattern-ExecutionDelayRate</siblingpattern>
    </relatedSiblingpatterns>
  </Relatedpatterns>
  <PatternConsequences>
</TransformPattern>
```

CSFPattern-ExecutionDelayRate

```
<TransformPattern>
  <PatternName>CSFPattern-ExecutionDelayRate</PatternName>
  <PatternGoal>Govern-delay-rate</PatternGoal>
  <PatternContext>pattern context--</PatternContext>
  <PatternPariticipants>paticipants:Req i; subpatterns; collaborated BP; etc.</PatternPariticipants>
  <PatternCollaboration>collaboration of participants</PatternCollaboration>
  <Relatedpatterns>
    <relatedParentpatterns>
      <parentpattern> Pattern-Performance</parentpattern>
    </relatedParentpatterns>
    <relatedSiblingpatterns>
      <siblingpattern>CSFPattern-ResponseDelayRate</siblingpattern>
    </relatedSiblingpatterns>
  </Relatedpatterns>
  <PatternConsequences>
</TransformPattern>
```

Figure 95 *Excerpt of Use Case Transform Pattern (subPattern-DelayRate)*

In our Use Case the organization of Governance Patterns pays attention on monitoring quality of Critical Success Factors. Due to in our Use Case Non Functional Property Classification registered 14 Critical Success Factors, there are 14 corresponding Goverannce Patterns (GPats) are registered in <GovernancePattern-UseCase.xml>. For example, Governance Patterns for CSF-ResponseDelayRate and CSF-ExecutionDelayRate are organized as following figure shows (see Figure 96):

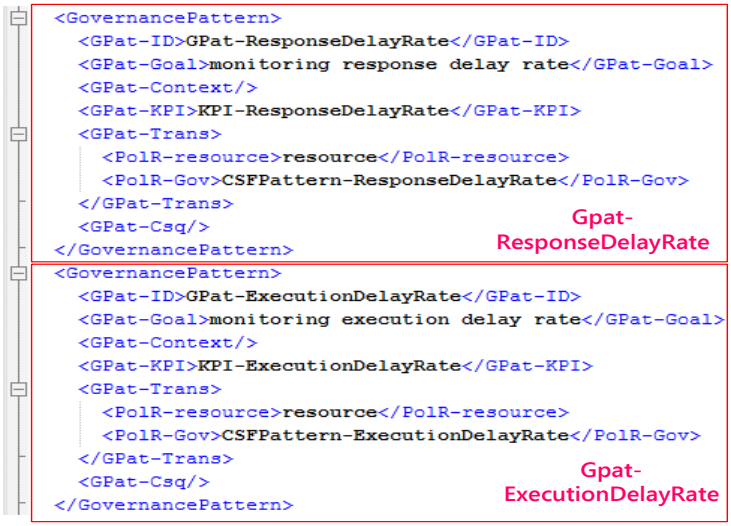


Figure 96 *Excerpt of Use Case - Transform Pattern (Four Group Pattern)*

3. Use Case – Implementation of Monitoring Policy Rule's Generation

Generated Root-PolRs for MonReq 2 are organized as:

- PolR 3 = ("PolR 3","count response delay time", "Resource(Standard Delivery BP, BPL, white-box)", "", "Gov(ResponseDelayRate, count delay time out of measured time)", "GPat-RespDelay");
- PolR 4 = ("PolR 4", "count execution delay time", "Resource(Standard Delivery BP, BPL, white-box)", "", "Gov(ExecutionDelayRate, count delay time out of measured time)", "GPat-ExecDelay")

All propagated monitoring policy rules for MonReq1 a organized as following:

- PolR 1_3 = ("PolR 1_3", "count response delay time", "Resource(Task_E1, BPL, white-box)", "", "Gov(response delay time, count delay time out of measured time)", "GPat-RespDelay")

- PolR 1_4 = (“PolR 1_4”, “count response delay time”, “Resource(Task_E2, BPL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 1_5 = (“PolR 1_5”, “count response delay time”, “Resource(Task_E3, BPL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 1_6 = (“PolR 1_6”, “count response delay time”, “Resource(Task_E4, BPL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 1_7 = (“PolR 1_7”, “count response delay time”, “Resource(Service_1, BSL, black-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 1_8 = (“PolR 1_8”, “count response delay time”, “Resource(Service_E2, BSL, black-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 1_9 = (“PolR 1_9”, “count response delay time”, “Resource(Service_E3, BSL, black-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 1_10 = (“PolR 1_10”, “count response delay time”, “Resource(Service_E4, BSL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 1_11 = (“PolR 1_11”, “count response delay time”, “Resource(OP_E3-1, BSL, black-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 1_12 = (“PolR 1_12”, “count response delay time”, “Resource(OP_E3-2, BSL, black-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 1_13 = (“PolR 1_13”, “count response delay time”, “Resource(OP_E4-1, BSL, black-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 1_14 = (“PolR 1_14”, “count response delay time”, “Resource(OP_E4-2, BSL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 1_15 = (“PolR 1_15”, “count response delay time”, “Resource(Inf_E4-2, BSL, black-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 2_3 = (“PolR 2_3”, “count execution delay time”, “Resource(Task_E1, BPL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)

- PolR 2_4 = (“PolR 2_4”, “count execution delay time”, “Resource(Task_E2, BPL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 2_5 = (“PolR 2_5”, “count execution delay time”, “Resource(Task_E3, BPL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 2_6 = (“PolR 2_6”, “count execution delay time”, “Resource(Task_E4, BPL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 2_7 = (“PolR 2_7”, “count execution delay time”, “Resource(Service_1, BSL, black-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 2_8 = (“PolR 2_8”, “count execution delay time”, “Resource(Service_E2, BSL, black-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 2_9 = (“PolR 2_9”, “count execution delay time”, “Resource(Service_E3, BSL, black-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 2_10 = (“PolR 2_10”, “count execution delay time”, “Resource(Service_E4, BSL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 2_11 = (“PolR 2_11”, “count execution delay time”, “Resource(OP_E3-1, BSL, black-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 2_12 = (“PolR 2_12”, “count execution delay time”, “Resource(OP_E3-2, BSL, black-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 2_13 = (“PolR 2_13”, “count execution delay time”, “Resource(OP_E4-1, BSL, black-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 2_14 = (“PolR 2_14”, “count execution delay time”, “Resource(OP_E4-2, BSL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 2_15 = (“PolR 2_15”, “count execution delay time”, “Resource(Inf_E4-2, BIL, black-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)

In the similar way for MonReq2 propagated PolRs can be organized as following.

- PolR 3_3 = (“PolR 3_3”, “count response delay time”, “Resource(Task 1, BPL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)

- PolR 3_4 = (“PolR 3_4”, “count response delay time”, “Resource(Task 2, BPL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 3_5 = (“PolR 3_5”, “count response delay time”, “Resource(Task 3, BPL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 3_6 = (“PolR 3_6”, “count response delay time”, “Resource(Task 4, BPL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 3_7 = (“PolR 3_7”, “count response delay time”, “Resource(Service_1, BSL, black-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 3_8 = (“PolR 3_8”, “count response delay time”, “Resource(Service_S2, BSL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 3_9 = (“PolR 3_9”, “count response delay time”, “Resource(Service_S3, BSL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 3_10 = (“PolR 3_10”, “count response delay time”, “Resource(Service_S4, BSL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 3_11 = (“PolR 3_11”, “count response delay time”, “Resource(OP S2-1, BSL, black-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 3_12 = (“PolR 3_12”, “count response delay time”, “Resource(OP_S2-2, BSL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 3_13 = (“PolR 3_13”, “count response delay time”, “Resource(OP_S3-1, BSL, black-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 3_14 = (“PolR 3_14”, “count response delay time”, “Resource(OP_S3-2, BSL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 3_15 = (“PolR 3_15”, “count response delay time”, “Resource(OP_S4-1, BSL, black-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 3_16 = (“PolR 3_16”, “count response delay time”, “Resource(OP_S4-2, BSL, white-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)

- PolR 3_17 = (“PolR 3_17”, “count response delay time”, “Resource(Inf_S2-2, BIL, black-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 3_18 = (“PolR 3_18”, “count response delay time”, “Resource(Inf_S3-2, BIL, black-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 3_19 = (“PolR 3_19”, “count response delay time”, “Resource(Inf_S4-2, BIL, black-box)”, “”, “Gov(response delay time, count delay time out of measured time)”, “GPat-RespDelay”)
- PolR 4_3 = (“PolR 4_3”, “count execution delay time”, “Resource(Task_1, BPL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_4 = (“PolR 4_4”, “count execution delay time”, “Resource(Task_2, BPL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_5 = (“PolR 4_5”, “count execution delay time”, “Resource(Task_3, BPL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_6 = (“PolR 4_6”, “count execution delay time”, “Resource(Task_4, BPL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_7 = (“PolR 4_7”, “count execution delay time”, “Resource(Service_1, BSL, black-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_8 = (“PolR 4_8”, “count execution delay time”, “Resource(Service_S2, BSL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_9 = (“PolR 4_9”, “count execution delay time”, “Resource(Service_S3, BSL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_10 = (“PolR 4_10”, “count execution delay time”, “Resource(Service_S4, BSL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_11 = (“PolR 4_11”, “count execution delay time”, “Resource(OP_S2-1, BSL, black-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_12 = (“PolR 4_12”, “count execution delay time”, “Resource(OP_S2-2, BSL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)

- PolR 4_13 = (“PolR 4_13”, “count execution delay time”, “Resource(OP_S3-1, BSL, black-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_14 = (“PolR 4_14”, “count execution delay time”, “Resource(OP_S3-2, BSL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_15 = (“PolR 4_15”, “count execution delay time”, “Resource(OP_S4-1, BSL, black-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_16 = (“PolR 4_16”, “count execution delay time”, “Resource(OP_S4-2, BSL, white-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_17 = (“PolR 4_17”, “count execution delay time”, “Resource(Inf_S2-2, BIL, black-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_18 = (“PolR 4_18”, “count execution delay time”, “Resource(Inf_S3-2, BIL, black-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)
- PolR 4_19 = (“PolR 4_19”, “count execution delay time”, “Resource(Inf_S4-2, BIL, black-box)”, “”, “Gov(execution delay time, count delay time out of measured time)”, “GPat-ExecDelay”)

Standard Delivery BP has been decomposed into 10 monitored nodes (see Figure 97): Service 1, Operation S2-1, Operation S2-2, Operation S3-1, Operation S3-2, Operation S4-1, Operation S4-2, Inf S2-2, Inf S3-2, Inf S4-2.

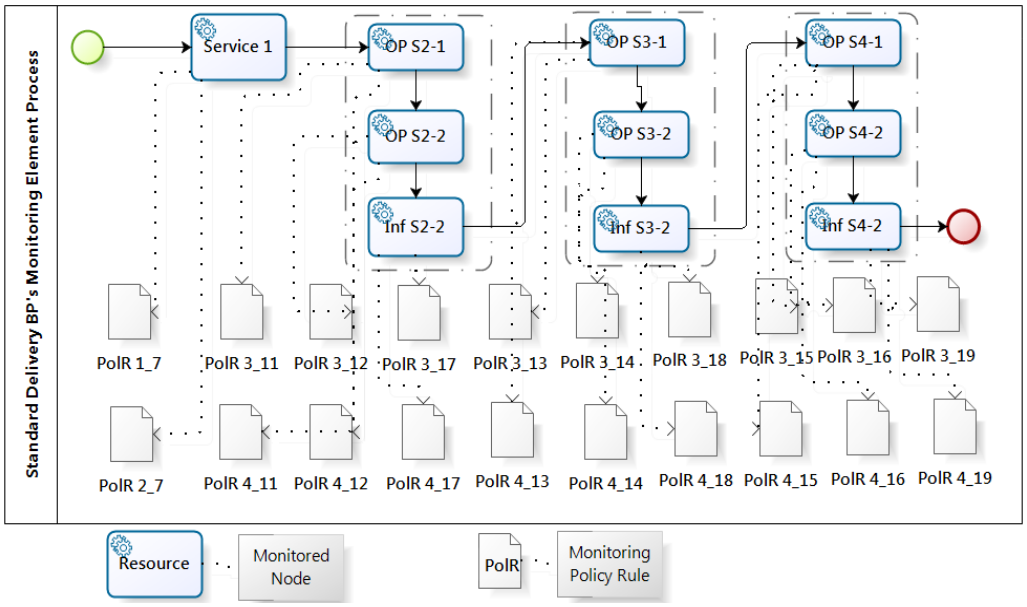


Figure 97 *Standard Delivery BP's Governance Elements (Process/Monitoring Policy Rules)*

Standard Delivery BP's Key Performance Indicators are defined as following (see Figure 98):

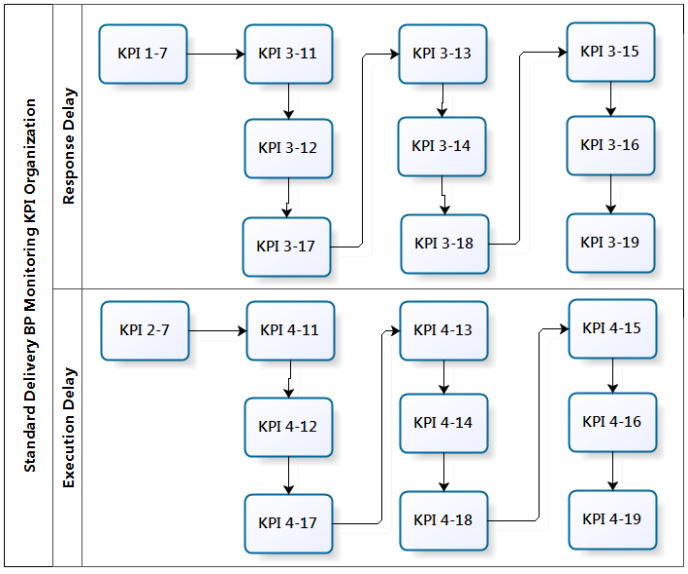


Figure 98 *Standard Delivery BP's Monitoring KPI Organization*

Required Key Performance Indicators are listed below:

- KPI 1-7 = (“KPI 1-7”, “monitor response delay”, “resource (Service 1/black-box/BSL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 1-8 = (“KPI 1-8”, “monitor response delay”, “resource (Service E2/black-box/BSL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 1-11 = (“KPI1-11”, “monitor response delay”, “resource (OP E3-1/black-box/BSL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 1-12= (“KPI 1-12”, “monitor response delay”, “resource (OP E3-2/black-box/BSL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 1-13 = (“KPI 1-13”, “monitor response delay”, “resource (OP E4-1/black-box/BSL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 1-14 = (“KPI 1-14”, “monitor response delay”, “resource (OP E4-2/black-box/BSL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 1-15 = (“KPI 1-15”, “monitor response delay”, “resource (Inf E4-2/black-box/BIL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 2-7 = (“KPI 2-7”, “monitor execution delay”, “resource (Service 1/black-box/BSL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 2-8 = (“KPI 2-8”, “monitor execution delay”, “resource (Service E2/black-box/BSL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 2-11 = (“KPI 2-11”, “monitor execution delay”, “resource (OP E3-1/black-box/BSL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 2-12 = (“KPI 2-12 ”, “monitor execution delay”, “resource (OP E3-2/black-box/BSL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 2-13 = (“KPI 2-13 ”, “monitor execution delay”, “resource (OP E4-1/black-box/BSL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 2-14 = (“KPI 2-14 ”, “monitor execution delay”, “resource (OP E4-2/black-box/BSL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 2-15 = (“KPI 2-15”, “monitor execution delay”, “resource (Inf E4-2/black-box/BIL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 1-7 = (“KPI 1-7”, “monitor response delay”, “resource (Service 1/black-box/BSL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 3-11 = (“KPI 3-11”, “monitor response delay”, “resource (OP S2-1/black-box/BSL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 3-12 = (“KPI 3-12”, “monitor response delay”, “resource (OP S2-2/black-box/BSL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 3-13= (“KPI 3-13”, “monitor response delay”, “resource (OP S3-1/black-box/BSL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 3-14 = (“KPI 3-14”, “monitor response delay”, “resource (OP S3-2/black-box/BSL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 3-15 = (“KPI 3-15”, “monitor response delay”, “resource (OP S4-1/black-box/BSL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 3-16 = (“KPI 3-16”, “monitor response delay”, “resource (OP S4-2/black-box/BSL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 3-17 = (“KPI 3-17”, “monitor response delay”, “resource (Inf S2-2/black-box/BIL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 3-18 = (“KPI 3-18”, “monitor response delay”, “resource (Inf S3-2/black-box/BIL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 3-19 = (“KPI 3-19”, “monitor response delay”, “resource (Inf S4-2/black-box/BIL)”, “CSF (Response delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 2-7 = (“KPI 2-7”, “monitor execution delay”, “resource (Service 1/black-box/BSL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 4-11 = (“KPI 4-11”, “monitor execution delay”, “resource (OP S2-1/black-box/BSL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

- KPI 4-12 = (“KPI 4-12”, “monitor execution delay”, “resource (OP S2-2/black-box/BSL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 4-13 = (“KPI 4-13”, “monitor execution delay”, “resource (OP S3-1/black-box/BSL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 4-14 = (“KPI 4-14 ”, “monitor execution delay”, “resource (OP S3-2/black-box/BSL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 4-15 = (“KPI 4-15 ”, “monitor execution delay”, “resource (OP S4-1/black-box/BSL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 4-16 = (“KPI 4-16 ”, “monitor execution delay”, “resource (OP S4-2/black-box/BSL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 4-17 = (“KPI 4-17”, “monitor execution delay”, “resource (Inf S2-2/black-box/BIL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 4-18 = (“KPI 4-18”, “monitor execution delay”, “resource (Inf S3-2/black-box/BIL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);
- KPI 4-19 = (“KPI 4-19”, “monitor execution delay”, “resource (Inf S4-2/black-box/BIL)”, “CSF (Execution delay rate, count delay time out of measured time)”, “Output(distance, data, Timestamp)”, “consequence{}”);

4. Composition Algorithms

I. Price:

Description: price is the fee that the service consumer is expected to pay for using a given service.

Data Structure: It includes two major parameters: Mode and Unit. Mode defines the way the customer is charged, such as per bill or a certain time period or a certain quantity. Unit defines how much the customer will pay and in which currency for each mode.

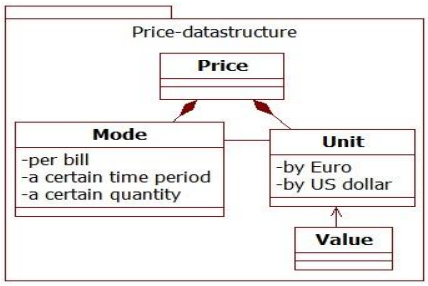


Figure 99 *NFP: Price's data structure*

Algorithms are defined as (see Figure 100):

Algorithm 1: Price.Value= $\sum_{i=1}^{n} price_i$ (Equation 86) is used for three composition types (viz. Sequence, Concurrency, Loop);

Algorithm 2: Price.Value= max (price $_1$ … price $_n$) (Equation 87) is used for the Conditional Branching composition type.

```
<NFP>
  <name>Cost</name>                          Cost Group
  <Elements>
  <Measurement>unit.value*mode.value</Measurement>
  <Algorithm>
  <ParenetNFP>none</ParenetNFP>
  <ChildNFP>Price</ChildNFP>
</NFP>                                       Definition of Price
<NFP>
  <name>Price</name>
  <Elements>
    <Element>unit.value</Element>
    <Element>payment.mode</Element>
    <Element>mode.value</Element>
  </Elements>
  <Measurement>unit.value*paymentmode.value</Measurement>
  <Algorithm>
    <AggregationAlgorithem>
      <Type>Sequence</Type>
      <Algorithm>price.value = Σ price i</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Concurrency</Type>
      <Algorithm>price.value = Σ price i</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Loop</Type>
      <Algorithm>price.value = Σ price i</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Conditional Branching</Type>
      <Algorithm>price.value = max ( price i )</Algorithm>
    </AggregationAlgorithem>
  </Algorithm>
  <ParenetNFP>Cost</ParenetNFP>
  <ChildNFP>none</ChildNFP>
</NFP>
```

Figure 100 *Definition of NFP-Price*

II. Time Relevant Non Functional Properties: Response Time and Execution Time

Description: Time is a common measure of performance and quality of Non Functional Property. This Time Relevant aggregation algorithm is used to aggregate Response time, Delay time or any time relevant Non Functional Properties.

Data structure: It includes two elements: unit and value. Unit defines the unit time interval for aggregate the time relevant Non Functional Property, value is the result of this particular Non Functional Property's quality in a given time interval unit (see Figure 101).

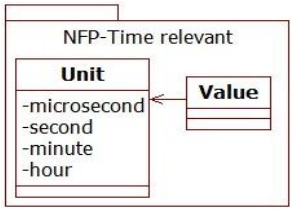


Figure 101 *NFP: Time Related NFP's data structure*

Algorithms are defined as follow (see Figure 102):

Algorithm 1: seq.Value= $\sum_{i=1}^{n} unit.value_i$ (Equation 88) is used for Sequence and Loop composition types;

Algorithm 2: max (unit.Value$_1$ … unit.Value $_n$) (Equation 89) is used for Concurren-cy composition type;

Algorithm 3: max (worst (unit.value)) + min (best (unit.value)) + avg(avg(unit.value)) (Equation 90) is used for Conditional Branching composition type.


```
<NFP>
  <name>Performance</name>
  <Elements>
  <Measurement>unit.value</Measurement>
  <Algorithm>
  <ParenetNFP>none</ParenetNFP>
  <ChildNFP>Response Time</ChildNFP>
  <ChildNFP>Execution Time</ChildNFP>
  <ChildNFP>Availability Rate</ChildNFP>
  <ChildNFP>Delay Rate</ChildNFP>
</NFP>
<NFP>
  <name>Response Time</name>
  <Elements>
    <Element>unit.value</Element>
  </Elements>
  <Measurement>unit.value</Measurement>
  <Algorithm>
    <AggregationAlgorithem>
      <Type>Sequence</Type>
      <Algorithm>responsetime.value = ∑ unit.value i</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Concurrency</Type>
      <Algorithm>max ( unit.value i )</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Loop</Type>
      <Algorithm>responsetime.value = ∑ unit.value i</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Conditional Branching</Type>
      <Algorithm>max ( worst ( unit.value ) + min ( unit.value ) + avg ( avg ( unit.value ) ) )</Algorithm>
    </AggregationAlgorithem>
  </Algorithm>
  <ParenetNFP>Performance</ParenetNFP>
  <ChildNFP>none</ChildNFP>
</NFP>
```

Figure 102 *Definition of Response time*

III. Reputation:

Description: reputation is an overall quality of a given service judged by previous service consumers.

Data structure: It has two elements: Ranking level and Comment Confidence. Ranking level defines a given service's reputation (from level 0 to level 5, the higher number being the better reputation). Comment Confidence defines how confident the consumer is about the given ranking level.

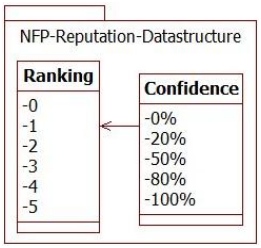


Figure 103 *NFP: Reputation's data structure*

The **Algorithm** implements: Reputation.Value = Ranking.Value * Confidence.Value (Equation 91)

Reputation can be expressed as Best (Reputation.Value), Worst (Reputation.Value) and Avg (Reputation.Value) used for all four composition types (see Figure 104).

```
120   <name>Maintainability</name>
121   <Elements>
122     <Element>unit.value</Element>
123     <Element>mode.value</Element>
124   </Elements>
125   <Measurement>unit.value</Measurement>
126   <Algorithm>
144   <ParenetNFP>none</ParenetNFP>
145   <ChildNFP>Reputation</ChildNFP>
146   <ChildNFP>Reliability</ChildNFP>
147   <ChildNFP>Usability</ChildNFP>
148   <ChildNFP>Accuracy</ChildNFP>
149 </NFP>
```

NFP Group: Maintainability

Definition of Reputation

```
150 <NFP>
151   <name>Reputation</name>
152   <Elements>
153     <Element>ranking</Element>
154     <Element>confidence</Element>
155   </Elements>
156   <Measurement>reputation.value=ranking*confidence</Measurement>
157   <Algorithm>
158     <AggregationAlgorithem>
159       <Type>Sequence</Type>
160       <Algorithm>best = max ( reputation.value )</Algorithm>
161       <Algorithm>worst = min ( reputation.value )</Algorithm>
162       <Algorithm>avg = avg (reputation.value)</Algorithm>
163     </AggregationAlgorithem>
164     <AggregationAlgorithem>
165       <Type>Concurrency</Type>
166       <Algorithm>best = max ( reputation.value )</Algorithm>
167       <Algorithm>worst = min ( reputation.value )</Algorithm>
168       <Algorithm>avg = avg (reputation.value)</Algorithm>
169     </AggregationAlgorithem>
170     <AggregationAlgorithem>
171       <Type>Loop</Type>
172       <Algorithm>best = max ( reputation.value )</Algorithm>
173       <Algorithm>worst = min ( reputation.value )</Algorithm>
174       <Algorithm>avg = avg (reputation.value)</Algorithm>
175     </AggregationAlgorithem>
176     <AggregationAlgorithem>
177       <Type>Conditional Branching</Type>
178       <Algorithm>best = max ( reputation.value )</Algorithm>
179       <Algorithm>worst = min ( reputation.value )</Algorithm>
180       <Algorithm>avg = avg (reputation.value)</Algorithm>
181     </AggregationAlgorithem>
182   </Algorithm>
183   <ParenetNFP>Performance</ParenetNFP>
184   <ChildNFP>none</ChildNFP>
185 </NFP>
```

Figure 104 *Definition of NFP-Reputation*

IV. Security Relevant Non Functional Properties:

Description: Protective measures that ensure the Service's inviolability state. Security relevant Non Functional Properties include Data Privacy, Certification, Encryption, Authentication, Non-Repudiation, Protection, etc. (see Figure 106)

Data structure: It has two elements: Security Measurement and Security Mark. Security Measurement defines the specific Security mechanism. The Security Mark indicates whether this service has to run security mechanism or not. The value of Security Mark can be 0 or 1, 0 means this service does not execute the required Security mechanism. We take this is an abnormal state. 1 means this service executed the required security mechanism which is a normal state for our governance architecture.

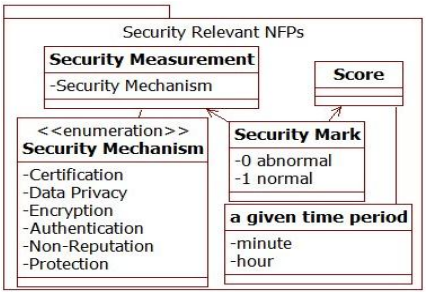


Figure 105 *NFP: Security Relevant NFP's Data Structure*

The **aggregation algorithm** of Resources security is complex. It cannot be calculate as other Non Functional Properties. To aggregate the security of all involved Resources, we grade the comprehensive security situation in a given time period by taking all context into account. Then we pay attention on the quality of business process. We give the score to security value, Security.score= [1 to 10] the highest score means the most satisfied security result.

```
<NFP>
  <name>security</name>
  <Elements>
    <Element>mechanism</Element>
    <Element>score</Element>
    <Element>time period</Element>
  </Elements>
  <Measurement>security.value= mechanism.score</Measurement>
  <Algorithm>
    <AggregationAlgorithem>
      <Type>Sequence</Type>
      <Algorithm>best = max ( security.value )</Algorithm>
      <Algorithm>worst = min ( security.value )</Algorithm>
      <Algorithm>avg = avg ( security.value)</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Concurrency</Type>
      <Algorithm>best = max ( security.value )</Algorithm>
      <Algorithm>worst = min ( security.value )</Algorithm>
      <Algorithm>avg = avg ( security.value)</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Loop</Type>
      <Algorithm>best = max ( security.value )</Algorithm>
      <Algorithm>worst = min ( security.value )</Algorithm>
      <Algorithm>avg = avg ( security.value)</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Conditional Branching</Type>
      <Algorithm>best = max ( security.value )</Algorithm>
      <Algorithm>worst = min ( security.value )</Algorithm>
      <Algorithm>avg = avg ( security.value)</Algorithm>
    </AggregationAlgorithem>
  </Algorithm>
  <ParenetNFP>non</ParenetNFP>
  <ChildNFP>none</ChildNFP>
</NFP>
```

Figure 106 *Definition of NFP-Security*

V. Rate Relevant Non Functional Properties:

Description: It can be expressed as a percentage of the expected results out of the total number of measured results in a given time period. It is a common method to show the quality of Non Functional Properties, such as Delay Rate in a given time period, Availability Rate in a given time period, etc.

5.1 Availability Rate:

Description: It is associate to the probability that a required resource can be accessed by Business Decision Maker in a given time period. The opposite Non Functional Property is Downtime Rate. It is impact the business performance directly. A poor availability result can cause bad reputation and a loss of business opportunity.

Data structure: It includes three elements: Downtime, measured time and a given time period.

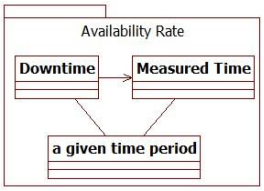


Figure 107 *NFP: Availability's Data Structure*

The **Algorithm** is built according to the following equation:

Availability Rate in a given time period: Availability = [1-(Down Time/Measured Time)]*100% (Equation 92);

Algorithm 1: $\prod_{i=1}^{n} Availability_i$ (Equation 93) is used for Sequence, Concurrency and Loop composition types;

Algorithm 2: Min (Availability $_i$ … Availability $_n$) (Equation 94) is used for Conditional Branching. In order to guarantee the quality of business process, we take the minimum availability of involved Resources as the Conditional Branching type's availability.

```
<NFP>
  <name>Availability Rate</name>
  <Elements>
    <Element>number of down time</Element>
    <Element>number of measured time</Element>
  </Elements>
  <Measurement>availbility.value=[ 1-( number of down time/ number of measured time )]*100%</Measurement>
  <Algorithm>
    <AggregationAlgorithem>
      <Type>Sequence</Type>
      <Algorithm>availbility.value = product ( unit.value i...n)</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Concurrency</Type>
      <Algorithm>availbility.value = product ( unit.value i...n)</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Loop</Type>
      <Algorithm>availbility.value = product ( unit.value i...n)</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Conditional Branching</Type>
      <Algorithm>min ( unit.value i...n )</Algorithm>
    </AggregationAlgorithem>
  </Algorithm>
  <ParenetNFP>Performance</ParenetNFP>
  <ChildNFP>none</ChildNFP>
</NFP>
```

Figure 108 *Definition of NFP-Availability*

5.2 Delay Rate:

Description: It is the percentage of delayed service's quantity out of total invoked service's quantity in a given time period.

Data Structure: it includes five elements: time unit, a given time period, amount of delayed time, amount of measured time and value.

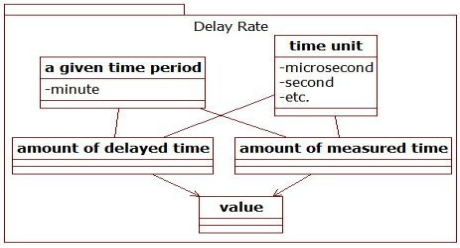


Figure 109 *NFP: Delay Rate Data Structure*

The **Algorithm** defined Figure 110 presents the Delay Rate in a given time period: DelayRate = (delayed time/ measured time)*100% (Equation 95)

Algorithm 1: $[1- (\prod_{i=1}^{n} DelayRate\ i)]*100\%$ (Equation 96) is used for Sequence, Concurrency and Loop composition types;

Algorithm 2: Max ($DelayRate_i$ … $DelayRate_n$) (Equation 97) is used for Conditional Branching type. In order to guarantee the quality of BaaS, we take the maximum delay rate of involved Resources as the Conditional Branching type's delay rate.

```
<NFP>
  <name>Delay Rate</name>
  <Elements>
    <Element>number of delayed time</Element>
    <Element>number of measured time</Element>
  </Elements>
  <Measurement>delayrate.value=( number of delayed time/ number of measured time )*100%</Measurement>
  <Algorithm>
  <ParenetNFP>
  <name>Performance</name>
  </ParenetNFP>
  <ChildNFP>
  <name>Response Delay Rate</name>
  <name>Execution Delay Rate</name>
  </ChildNFP>
</NFP>
<NFP>
  <name>Response Delay Rate</name>
  <Elements>
    <Element>number of delayed time</Element>
    <Element>number of measured time</Element>
  </Elements>
  <Measurement>delayrate.value=( number of delayed time/ number of measured time )*100%</Measurement>
  <Algorithm>
    <AggregationAlgorithem>
      <Type>Sequence</Type>
      <Algorithm>delayrate.value = [1- ( product ( unit.value i...n))]*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Concurrency</Type>
      <Algorithm>delayrate.value = [1- ( product ( unit.value i...n))]*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Loop</Type>
      <Algorithm>delayrate.value = [1- ( product ( unit.value i...n))]*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Conditional Branching</Type>
      <Algorithm>max ( unit.value i...n )</Algorithm>
    </AggregationAlgorithem>
  </Algorithm>
  <ParenetNFP>
  <name>Delay Rate</name>
  </ParenetNFP>
  <ChildNFP>none</ChildNFP>
</NFP>
```

Delay Rate has two child NFPs

Figure 110 *Definition of NFP-Response Delay Rate*

```
<NFP>
  <name>Execution Delay Rate</name>
  <Elements>
    <Element>number of delayed time</Element>
    <Element>number of measured time</Element>
  </Elements>
  <Measurement>delayrate.value=( number of delayed time/ number of measured time )*100%</Measurement>
  <Algorithm>
    <AggregationAlgorithem>
      <Type>Sequence</Type>
      <Algorithm>delayrate.value = [1- ( product ( unit.value i...n))]*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Concurrency</Type>
      <Algorithm>delayrate.value = [1- ( product ( unit.value i...n))]*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Loop</Type>
      <Algorithm>delayrate.value = [1- ( product ( unit.value i...n))]*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Conditional Branching</Type>
      <Algorithm>max ( unit.value i...n )</Algorithm>
    </AggregationAlgorithem>
  </Algorithm>
  <ParenetNFP>
  <name>Delay Rate</name>
  </ParenetNFP>
  <ChildNFP>none</ChildNFP>
</NFP>
```

Figure 111 *Definition of NFP-Execution Delay Rate*

5.3 Reliability:

Description: It is the consistency of a resource's execution under given conditions for a given interval of time. A high reliability means a resource requires less debugging and maintenance.

Data structure: It includes three elements: number of successful execution, total number of execution and a given time period.

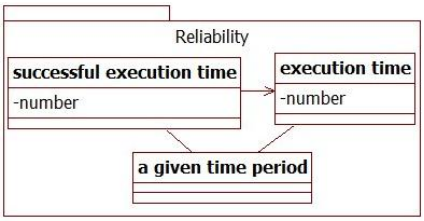


Figure 112 *NFP: Reliability's Data Structure*

The **Algorithm** given Figure 113 computes reliability is Reliability = (number of successful executions / Total number of execution) (Equation 98) in a given time period.

Algorithm 1: $\prod_{i=1}^{n} Reliability_i$ (Equation 99) is used for Sequence, Concurrency and Loop composition types.

Algorithm 2: Min ($Reliability_i$ … $Reliability_n$) (Equation 100) is used for Conditional Branching type. In order to guarantee the quality of BaaS, we take the worst reliability of involved services as the Conditional Branching composition's reliability.

```
<NFP>
  <name>Reliability</name>
  <Elements>
    <Element>number of successful time</Element>
    <Element>number of executed time</Element>
  </Elements>
  <Measurement>reliability=( number of successful time/ number of executed time )*100%</Measurement>
  <Algorithm>
    <AggregationAlgorithem>
      <Type>Sequence</Type>
      <Algorithm>reliability =  product ( unit.value i...n)*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Concurrency</Type>
      <Algorithm>reliability  = product ( unit.value i...n)*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Loop</Type>
      <Algorithm>reliability  = product ( unit.value i...n)*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Conditional Branching</Type>
      <Algorithm>reliability = min ( unit.value i...n )</Algorithm>
    </AggregationAlgorithem>
  </Algorithm>
  <ParenetNFP>
    <name>Performance</name>
  </ParenetNFP>
  <ChildNFP>none</ChildNFP>
</NFP>
```

Figure 113 *Definition of NFP-Reliability*

VI. Usability:

Description: It is the capability of the resource to be understood, learned, used in a given time period by a certain number of consumers.

Data structure: It includes three elements: Score, Utilization and Time period. Score indicates the feedback from consumers from 0 to 5, the higher scores means higher usability. Utilization defines how many consumers graded this service's usability. Time Period defines a given interval time.

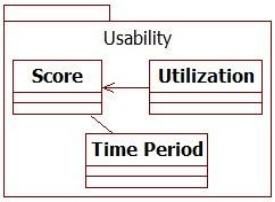


Figure 114 *NFP: Usability's data structure*

In the **Algorithm** (Figure 115): usability can be expressed as Usability.Best = Max (Usability $_i$ … Usability $_n$); Usability.Avg = Avg (Usability $_i$ … Usability $_n$); Usability.Worst = (Usability $_i$ … Usability $_n$).
(Equation 101)

Algorithm 1: Agg.usability = $\prod_{i=1}^{n} Usability_i$ (Equation 102) is used for Se-quence, Concurrency and Loop composition types.

Algorithm 2: Agg.usability = Usability.Worst (Equation 103) is used for Conditional Branching type.

```
<NFP>
  <name>Usability</name>
  <Elements>
    <Element>utilization</Element>
    <Element>score</Element>
  </Elements>
  <Measurement>usability.value=utilization.score</Measurement>
  <Algorithm>
    <AggregationAlgorithem>
      <Type>Sequence</Type>
      <Algorithm>usability =  product ( unit.value i...n)*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Concurrency</Type>
      <Algorithm>usability = product ( unit.value i...n)*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Loop</Type>
      <Algorithm>usability  = product ( unit.value i...n)*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Conditional Branching</Type>
      <Algorithm>usability = worst ( unit.value i...n )</Algorithm>
    </AggregationAlgorithem>
  </Algorithm>
  <ParenetNFP>
    <name>Performance</name>
  </ParenetNFP>
  <ChildNFP>none</ChildNFP>
</NFP>
```

Figure 115 *Definition of NFP- Usability*

VII. Accuracy:

Description: It is the ability for a resource to get results that meets the expected result precisely. Accuracy can be measured as a percentage of accuracy results out of the total measured results in a given time period.

Data structure: It includes five elements: Expected Results, Measured Results, Comparison Method, Value and Time Period. Expected Results defines the resource's expected outputs which are the accurate results. Measured Results defines the actual outputs of resource. Comparison Method defines the different ways to compare the actual results with expected results. Value defines the percentage of accurate results out of measured results. Time Period defines a given interval time to measure a resource's results.

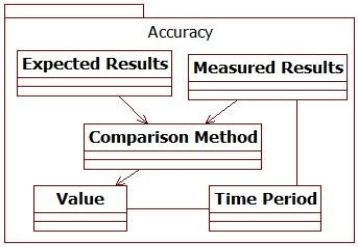


Figure 116 *NFP: Accuracy's Data structure*

The **Algorithm** computes: Accuracy = (1- unexpected results/measured results)*100%. It can be expressed as Accuracy. Best = Max (Accuracy $_i$ … Accuracy $_n$); Accuracy.Avg = Avg (Accuracy $_i$ … Accuracy $_n$); Accuracy.Worst = Min (Accuracy $_i$ … Accuracy $_n$). (Equation 104)

Algorithm 1: $\prod_{i=1}^{n} Accuracy_i$ (Equation 105)

is used for Sequence, Concurrency and Loop composition types.

Algorithm 2: Min (Accuracy $_i$ … Accuracy $_n$) (Equation 106)

is used for Conditional Branching type. In order to guarantee the quality of business process, we take the worst accuracy of involved services as the Conditional Branching composition's reliability.

```
<NFP>
  <name>Accuracy</name>
  <Elements>
    <Element>expected result</Element>
    <Element>measured result</Element>
    <Element>comparison method</Element>
    <Element>time period</Element>
  </Elements>
  <Measurement>accuracy=( 1- unexpected result / measured result )*100%</Measurement>
  <Algorithm>
    <AggregationAlgorithem>
      <Type>Sequence</Type>
      <Algorithm>accuracy =  product ( unit.value i...n)*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Concurrency</Type>
      <Algorithm>accuracy = product ( unit.value i...n)*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Loop</Type>
      <Algorithm>accuracy  = product ( unit.value i...n)*100%</Algorithm>
    </AggregationAlgorithem>
    <AggregationAlgorithem>
      <Type>Conditional Branching</Type>
      <Algorithm>accuracy = min ( unit.value i...n )</Algorithm>
    </AggregationAlgorithem>
  </Algorithm>
  <ParenetNFP>
    <name>Performance</name>
  </ParenetNFP>
  <ChildNFP>none</ChildNFP>
</NFP>
```

Figure 117 *Definition of NFP-Accuracy*