Linkage Mechanisms for component-based Services and IT Governance

Carsten Mueller
ITG Research
Nuremberg, Germany
carsten.mueller@itg-research.net

Abstract: Insufficient modularizing of services and applications leads to complexity and complication with IT Governance architectures. The problem is that the available IT Governance solutions on the market only offer limited dynamic exchangeable components and/or specialized services, in order to be able to react flexibly to changing conditions within a short time. IT Governance requirements, frameworks, processes and services are insufficiently supported based on the generalization and with that connected inflexibility. In this article an innovative Framework based on Linkage Mechanisms and component-based Services in the context of IT Governance is presented.

Key words: IT Governance, Component-based Software Engineering, Multi-Objective Optimization

1. Introduction

Information technology is among the main capital investments and operating expenditures being made by organizations today. Governance in the IT domain has the implication of keeping IT processes under control. The primary goal is to ensure that IT decisions are aligned with business goals, and that risks are managed. A control framework for governance provides a process and audit model. IT Governance (ITG) is specifying the decision rights and accountability framework to encourage desirable behaviour in using IT (Weill & Ross, 2004) ITG is a complex system. Each ITG implementation takes place in different conditions and circumstances (the ITG environment) determined by a large set of factors (e.g. company size or sector of economy the company is operating in). A static model (e.g. provided by Weill & Ross, 2004) of ITG and organization cannot adequately address these issues. Static ITG frameworks must be customized completely on account of changing environmental conditions and be delivered.

Two major frameworks for understanding ITG implementation are prominent in the literature. Firstly, Weill & Ross’ (2004) framework focuses on decision-making structures, alignment processes and communication approaches and provides a matrix of governance arrangements (business monarchy, IT monarchy, feudal, duopoly and anarchy) for ITG specific decisions. Secondly, Van Grembergen & De Haes (2009) framework highlights ITG arrangements of structures, processes and relational mechanisms. ITG structures focus on the roles and responsibilities of the IT/business committee, while ITG processes refer to the IT decision-making process and monitoring procedures. ITG relational mechanisms emphasise the active participation and collaboration of corporate executives, IT management and business management to facilitate the coordination of ITG structures and processes.

ITG concerns both the vertical allocation of decision rights between corporate headquarters and business units, and the horizontal allocation of decision rights between IT and other functional departments (Weill & Ross, 2004, Haes & Grembergen, 2009, Sambamurthy & Zmud, 1999 and Tiwana, 2009. Tiwana (2009) shows the allocation of decision rights for project management governance is mainly horizontal between IT and functional departments. On the other hand, the allocation of decision rights for ITG is mainly vertical between corporate headquarter and business unit (Sambamurthy & Zmud, 1999).

Effective ITG is crucial for an organisation to achieve its corporate performance goals. Evidence indicates that organizations that have implemented ITG generate 40 percent higher IT returns compared to those organizations who haven’t (Weill & Ross, 2004)

To implement ITG effectively, a set of ITG mechanisms is required that encourages behaviours congruent with the organisation’s mission, strategy, values, norms and culture (Weill & Ross, 2004). Organisations with ineffective ITG, by contrast, suffer due to inaccurate information quality, inefficient...
operating costs, runaway IT project costs (e.g., being over budget and under specification), loss of competitiveness, the demise of IT departments, or the organisation itself. To achieve effective ITG, an organisation needs to employ well-designed, well-understood, and transparent governance mechanisms.

Effective use of IT resources requires effective ITG structures. Effective ITG structures provide strategies that promote the effective management of IT resources so that these resources contribute to business value (Lainhart, 2001).

ITG consists of structures, processes, and relational mechanisms working together as one entity to ensure that IT and business objectives are aligned (Haes & Grembergen, 2009). A common theme found in literature is that effective ITG helps mitigate IT-related risks and gives reasonable assurance that the organisation's IT processes are consistently delivering the efficiency gains for which they are designed. The cornerstone of ITG is to provide an acceptable level of assurance that an organisation's strategic objectives are not jeopardised by IT failures (Spremic, 2009).

The problem is that the available ITG solutions on the market only offer limited dynamic exchangeable components and/or specialized services, in order to be able to react flexibly to changing conditions within a short time. An example is reporting solutions. An enterprise has different requirements regarding audit reporting. Due to administration and knowledge aspects it is in general not possible to uninstall the reporting applications and immediately install a new one.

At the run-time no dynamic (service) composition is possible for the optimization of (resource-) flows. Optimal (resource-) flows mean the fulfillment of ITG requirements and associated Quality-of-Service (QoS) by using a minimum of resources (e.g. costs).

The goal of this article is proposition of a framework for component-based ITG using Linkage Mechanisms to allow the determination of an optimum (resource-) flow based on Dynamic ITG Service Composition.
2. Interdisciplinary approach

In the interdisciplinary approach the areas ITG, Componentization and Patterns / Optimization to Linkage Mechanisms for component-based Services and ITG are combined. This combination is the basis for the implementation of the component-based ITG. The Linkage Mechanisms are the glue between the different disciplines and ensure the optimum (resource-)flow.

![Diagram showing the interdisciplinary approach]

3. Component and Service

Componentization reduces large-scale complexity, because the internals of each component are hidden. The traditional Software Engineering technology is based on components. Software development based on component composition is a critical technology of software reuse, which provides a kind of effective bottom-up way of using standard software components to structure system and has got a wide application.

Component-based Software Engineering (CBSE) denotes the process of building software by (re)using pre-built software components. CBSE is a reuse based approach to define, implement and compose loosely coupled independent components into systems. It is developed interconnecting building blocks, therefore, a high degree of reusability and modularity is achieved by this type of applications (Ioannidou & Payton, 2001). One main benefit of component-based software is the 'time-to-market', thus reducing the cost of developing the software (Lifang & Matta, Lowe, 2009).

In general, a component is a modular part of a system that encapsulates its content and whose manifestation is replaceable in the environment. It represents a logical unit, such as a class or package. The component is a piece of code that makes explicit its functionalities (interface) and its dependencies while hiding its internal structure and content. Components are expected to be deployed in containers providing the execution environment and an abstraction layer for their isolation from the operating system and infrastructure resources.

A software component is a unit of composition with contractually specified interfaces and explicit context dependencies. Additionally, a component is a software element that conforms to a component model and can be independently deployed and composed.

The term 'service' exists for hundreds of years along human history. When a person or a group performs some tasks to benefit another, this act or performance becomes a service. In order to design an appropriate service, in current service economy, service providers have to collaborate with their customers to co-create services for high value. Service is a discrete unit of functionality that is made available through a service contract.

The top part of the service is the service interface, and the bottom of the service is the service implementation. A service specifically separates the interface from the implementation.
Component services are declared through interface. Interface is the only information source for users to understand and use components, and also determines the constraint conditions that the system reuse components must be followed. Users understand the function of components through interface, which determines whether it fits their needs.

4. Service-oriented Architecture

Service-orientation is a promising paradigm to decompose inward-oriented organisational processes into outward-oriented ITG Service Components. With the aim of building large scale systems while reducing cost and dealing with complexity, dynamism, heterogeneity and uncertainty, autonomic computing principles have paved necessary foundations towards self-managing systems that are self-configuring, self-healing, self-optimizing, and self-protecting Salim et al. (2006). Supported by this architecture, services have become the basic blocks for building information systems from loosely-coupled elements. The binding of services into flexible compositions is recognized as a powerful paradigm for building distributed applications. This new paradigm has been adapted into Service Oriented Architecture (SOA).

The service-oriented computing paradigm, which provides an effective means of application abstraction, integration and reuse with its loosely-coupled architecture, presently has a dominant position in developing web-based information systems Loutas & Peristeras & Tarabanis (2001). In service-oriented software engineering, a component is evolving into a service with standard interfaces for communications and the ability to dynamically locate necessary services at runtime. It is as a natural progression from component-based software development, and as a means of integrating different component development frameworks. It is considered as advanced component-based architectures for the construction of distributed systems. SOA is seen as a continuum of different components at different levels of system abstraction, like the infrastructure, platform/middleware, and software viewpoints. A SOA is a software architecture using loosely coupled service providers. The SOA integrates them into a distributed computing system by means of service-oriented programming.

Service providers are made available as independent components that can be accessed without a priori knowledge of their underlying platform, implementation, and location. The client-server architecture separates a client from a server. SOA introduces a third component, a service registry. The registry allows the meta-operating system (not the requestor) to dynamically find service providers.

![Fig. 4: Elements of the Service-oriented Architecture](image)

A service in this context is defined as a behaviour that is provided by a component for use by any other component based on a network-addressable interface contract (generally identifying some capability provided by the service). A service stresses interoperability and is dynamically discovered and used. The service abstraction is used to specify access to resources in a unified way. How the actual service is implemented is hidden from the user through the service interface.

5. Framework for component-based IT Governance

The five ITG Components Request Management, Impact Management, Asset-/Change Management, Financial Management and Compliance Management are defined in the first inside shell. Each of these ITG Components fulfills a defined task area with specified (generic) ITG Services. The alignment follows a lifecycle-principle.
Seen from inside the second shell, the elements Linkage Mechanisms, Structure, Behaviour and Optimization are arranged. The Linkage Mechanisms use methods of optimization for the dynamic composition of ITG Services, in order to define an optimal (e.g. minimum cost for a given quality level or maximum suitability for a given cost level) resource flow. Each ITG Component has a defined structure and context-dependent and strategy-oriented properties. The Business-IT Alignment is based on the second shell.

In service-oriented software engineering, a component is evolving into a service with standard interfaces for communications and the ability to dynamically locate necessary services at runtime. It is as a natural progression from component-based software development, and as a means of integrating different component development frameworks. It is considered as advanced component-based architectures for the construction of distributed systems.

ITG Component Architecture is a set of significant decisions about the organization of an ITG Component Domain, the selection of the structural elements and their ITG Services by which the ITG Component Domain is composed, together with their behaviour as specified in the collaborations among these elements into progressively larger subsystems, and the architectural styles that guides this organization.

6. IT Governance Domain Analysis

Domain abstractions improve the reusability of components because they reduce the cognitive distance between the initial concept of a system and its final executable implementation. The basic idea of the domain analysis is to decompose the ITG environment into features in order to provide dynamic service composition and to facilitate the generation of linked ITG Components and ITG Services based on a selection of features.

A domain is described in a domain model, which is defined in a context described by a context model. A context model defines interactions between the candidate domain and external domains, together with any external constraints on these interactions. In addition to the context model, the domain model consists of many other (sub)models. This hierarchy is fundamental for the ITG Ontology which is used to model the elements (e.g. classes and relationships) and to determine similarities.
The developed mediator described in Chapter 8 adopts Information Retrieval similarity models such as vector space models, probabilistic models, and information theory-based models. Their underlying technique is semantic similarity measurement between ITG Services, either based on keywords or on the ITG Ontology.

ITG Domains support the definition of more than one ITG Service on the boundary of an ITG Component. This is done by defining a public ITG Domain and by linkage that ITG Domain to the owner ITG Domain of the main ITG Service. All ITG Services of that public ITG Domain then access the representation of the main ITG Service and are accessible by the stakeholder in the ITG environment.

The implementation of ITG Components follows a phase concept. In the first phase domain analysis, the specific requirements of a business will be analysed with regards to the ITG. Results of the domain analysis are relevant ITG Objects, ITG Services and their relation to each other. The phase domain analysis is supported by existing Frameworks (e.g. COBIT), and are taken over by the relevant ITG Objects and ITG Services.

The responsibility of the Domain Layer is the identification, analysis, and specification of ITG Feature and ITG Service Requirements from a specific ITG Domain.

![Fig. 6: IT Governance Domain Engineering](image)

It identifies the basic elements of the ITG Domain, organise an understanding of the relationships among these elements, and represent this understanding in a useful way.

Domain analysis in the context of ITG deals with identifying stakeholders and their objectives in a ITG Domain, defining selection criteria, identifying boundary conditions, characterising main common ITG Features and variants of the ITG Domain, determining relations to other ITG Domains, dividing the ITG Domain into Sub-Domains, acquiring ITG Domain information from experts, describing ITG Domain terminology, and building overall ITG Domain Models.

The ITG Domain denotes what the decisions should consider. It comprises four dimensional units goals, processes, people and technology.

The main goal of information analysis is to capture domain knowledge in the form of domain entities and links between them. The result of information analysis is the information model. Operational analysis results in the operational model. It represents how the ITG application based on ITG Component works and covers the links between ITG objects and ITG Services in the informational model and the ITG Features in the ITG Feature Model.

7. **Optimization with Evolutionary Algorithms**

Evolution is the theory postulating that all the various types of living organisms have their origin in other pre-existing types, and that the differences are due to modifications inherited through successive generations. It is the process which enables individuals or species in one generation to modify or improve in the next generation. The nature helps individuals to adapt to the changing environment through the process of evolution. In nature, species have, from a biological point of view, one main goal, which is to survive. As a result, almost all species are perfectly adapted to their surroundings, therefore maximising the chance of survival as a whole. A good search technique must find a good trade-off between exploration and exploitation in order to find a global optimum.

The basic Evolutionary Algorithm is composed of two processes. The first process is selection of individuals for the production of the next generation and the second process is manipulation of the
selected individuals to form the next generation by crossover and mutation techniques. The selection mechanism determines which individuals are chosen for mating (reproduction) and how many offspring each selected individual produces. Generally, crossover and mutation explore the search space, whereas selection reduces the search area within the population by discarding poor solutions.

Fig. 7: General scheme of an Evolutionary Algorithm

The mediator uses Evolutionary Algorithms in the optimisation process which is shown in Figure 8. The input for the optimization is the ITG Component Architecture and the Quality of Service Model. Based on Evolutionary Algorithms a feasible pareto-optimal ITG application configuration is determined and composed for workflow and/or application used in the corresponding ITG Domain.

Fig. 8: Mediator - Optimisation process

8. Mediator and Dynamic Service Composition

A component-based mediator model is proposed, because component software addresses the issues of adaptability and reusability. These qualities imply that (a) the behaviour of one component is changeable on the basis of different requirements and contexts, (b) several components can be composed to build new more powerful components, and (c) existing components can be replaced by new or advanced components. Another purpose is to introduce mediators when composing services together to make both of the requester and provider interfaces fit each other smoothly. Thus every mediator should have at least one input and/or output interface(s) connecting to the requester and/or the provider interface(s). A mediator component is defined as a software component, which takes part in a mediation process, has a contractual specified interface revealing its syntax and semantic (introspection). The composition of ITG Components using the mediator is based on interactions between ITG Components as a subset of behaviour in a system.

The ITG Mediator simplifies this ITG context by being the only ITG Component that is aware of the other ITG Components and is similar to a communication hub and accepts ITG Service Requests from
the five ITG Components focused on issues in the ITG Lifecycle. In the context of ITG components it is a semantic matchmaking system. Returning a ranked list of offers is a very important task for semantic matchmaking systems.

- The ITG Mediator selects metadata information of the corresponding ITG Components for the keyword-based search. An ITG Component with an associated ITG Services is linked within a context of ITG Requirements.
- The ITG Mediator is responsible for the Linkage Management and the Dynamic Composition of ITG Services.
- Based on ITG Service Keywords an ITG Service Request Query to the ITG Component Repository is carried out.
- The keyword-based search delivers - if applicable - the suitable ITG Services (the highest agreement between ITG Service Keywords and metadata information) for a Dynamic Composition by Linkage Mechanisms.

To synthesise a series of ITG Processes from a given Request, the ITG Mediator discovers an initial ITG Component which implements an ITG Service to perform the ITG Activity node in the Request. Next, the ITG Mediator discovers other ITG Components which provides necessary data to execute the ITG Service of the initial ITG Component and synthesises a partial ITG Process by interconnecting the output ITG Objects of the discovered ITG Components with the input ITG Objects of the initial ITG Component.

The ITG Mediator repeatedly discovers necessary ITG Components and expands an ITG Process until the ITG Process becomes executable, i.e. until all input ITG Objects of the ITG Components in the ITG Process are provided by the output ITG Objects of the other ITG Components in the ITG Process. Once it synthesised an executable ITG Process, the Mediator asks the ITG Component Compliance Management to check if the synthesised ITG Process semantically satisfies ITG Requirements.

9. Prototype and current status of implementation

In the following the current status of of the implementation is explained. Based on the long-year industrial experience a prototype was developed which is shown in Figure 6.

A stakeholder in the context of ITG requests support by ITG Components and linked ITG Services. The manual request by the stakeholder is – in this case – broken down into a request for the SAP Application Cost.

Based on keywords the request is parsed by the Mediator. The parsed keywords are added by information provided the ITG Component Compliance Management.

The mediator sends a service request to the component pool (repository) for query best suitable services regarding the keywords and possible additional constraints.

Found services are ranked and Linking Mechanisms are dynamically applied to compose a running workflow and/or application (in this case application to determine SAP Application Cost) to fulfil the requirements.

Currently the different algorithms are implemented and integrated in the basic framework for production.

Additionally the Evolutionary Algorithms and Service Recommender are optimized and an intensive sensitivity analysis is performed to determine the dependent parameters and effects on different ITG environment configurations. It is planned that the basic framework will be available for download this year.

For research updates please visit http://www.itg-research.net.
10. Conclusions and Future Perspectives

A Framework for component-based ITG was developed starting with the presentation of a problem from static ITG. The result is a generic, model-based, reusable and expandable building block for the component-based design of an ITG Domain. This building block is composed of process-oriented models, structured processes, strategies for optimization and application cases. It was shown how the architecture for the performance of a concrete ITG Domain is instrumented and in this way component-based ITG is performed.


The dynamic composition of ITG Components and corresponding ITG Services are centrally coordinated by a Mediator based on strategies and algorithms. The dynamic ITG Service composition, in the ITG context and ITG Components, was tested with different algorithms. Monte Carlo Method for ITG objective search spaces with maximum 50 elements (small and intermediate businesses) and Evolutionary Algorithms for ITG Objective Search Spaces with more than one hundred elements (enterprise) emerged as suitable algorithms. Every ITG Component and connected ITG objects are ordered into an ITG Domain.

Within a view of further research markets for component-based ITG are sketched. One such marketplace has brought together the economical function, the offer and the demand of ITG Components. Based on the availability and use of the Internet, it is possible and wise, to design and organize market places for ITG Components, principally as electronic market places.

References


JEL: M15