IT GOVERNANCE FRAMEWORKS AS METHODS

Matthias Goeken and Stefanie Alter
Frankfurt School of Finance and Management, IT-Governance-Practice-Network, Sonnemannstraße 9-11, Frankfurt

Keywords: IT Governance, IT Management, Method Engineering, Metamodel, COBIT (Control Objectives for Information and Related Technology).

Abstract: IT management of today only to a surprisingly low degree is based on sound methods. Rather it is considered still as an art attributed to the capabilities of individual managers. On the other hand, there is little academic support for the challenges of IT management. To fill this obvious gap, various best practice frameworks have been developed. These frameworks cover both support of management tasks and improvement of compliance with regulations. Both areas can be subsumed under the topic IT governance. Whereas the frameworks reflect best practice experience, they do not provide prove of e.g. completeness and coherence with respect to the application area they were designed for. To undertake a step in this direction we will explore, to what degree the well known IT governance framework COBIT may be classified as a method. By analyzing the underlying logical and semantically rich structure of this framework we gain insights on how to compare and integrate further frameworks with COBIT. We will conclude by giving indications on how we intend to implement IT governance frameworks into a toolset based on semantic nets.

1 INTRODUCTION

Unfortunately there is little academic guidance for the management of IT in general and for specific challenges, especially for business/IT alignment or risk management (Booth/Philip, 2005; p 395; Avison 2004, p 224). The better part of computer science research deals with system development and related issues. Given the fact that in enterprises usually a higher percentage of expenditure is spent on 'running IT' rather than on systems engineering and development of new systems, it is obviously critical to offer guidance for governance challenges, i.e. long term management and compliance with regulations.

Due to the fact, that there is a clear need for methodological support for the actual tasks and challenges of IT management and IT governance, it is surprising, that little attention is paid to these questions. Chan et al. (1997) and Reich/Benbasat (1996) censured researchers for the lack of effort put into evaluating e. g. how business and IT can be properly aligned, how IT related risks can be managed and how IT (Information Technology) can contribute to the overall value of the enterprise.

In recent years, there were some associations and public institutions like ISACA (Information Systems Audit and Control Association) and CCTA (Central Computer and Telecommunication Agency)/OGC (British Office of Government Commerce) which developed frameworks (e.g. COBIT and ITIL) to support management and governance of IT. These frameworks are well established in practice (KPMG 2004; ITGI 2006). However, these frameworks are lacking of theoretical foundation which would allow conclusion with regard to e.g. completeness and coherence.

This paper takes steps towards the theoretical foundation of best practice frameworks by proposing to model them as methods. We will discuss frameworks as 'methods' for IT management. Due to the fact, that 'method engineering' and 'method construction' can be seen as the core of a design science oriented information systems research (Hevener 2004; Braun et al. 2005), we argue, that methods for IT management should be based on these well elaborated approaches.

We therefore derive an adapted method metamodel for IT governance from a discussion of well established method engineering approaches (part 2). These we propose to extend, in order to capture management and governance related aspects. Furthermore we present a method metamodel of COBIT, the popular governance framework of the ISACA (part 3). This will be compared with the theoretically derived method metamodel. Afterwards

Goeken M. and Alter S. (2008).
IT GOVERNANCE FRAMEWORKS AS METHODS.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - ISAS*, pages 331-338
DOI: 10.5220/0001705403310338
Copyright © SciTePress

we discuss the advantages of this approach and present some research in progress – e.g. a prototype with which we modeled COBIT.

2 METHOD ENGINEERING

In IS/CS research, the notion of the term ‘method’ is mainly related to system development and similar disciplines. In general ‘method’ is being used as a quite generic term – coming from Greece “méthodos” (way/procedure). Henceforth we will relate it to IT management only. The following section will give a brief introduction of the terms method and method engineering and will provide the foundation for the discussion about the notion of methods in IT management / IT governance.

Methods are usually comprehensive approaches, structuring the whole process of system development. Therefore, a method typically has a holistic scope and covers all tasks and activities, necessary to plan, design and implement a system. Although there is much research on the topics of methods and method engineering (e.g. Ralyté et al. 2007), a generally accepted definition of ‘method’ including its components is still missing. As a result, ‘process model’, ‘lifecycle model’ as well as ‘technique’ are often used synonymously or interchangeable.

An often cited definition has been coined by Brinkkemper: „A method is an approach to perform a systems development project, based on a specific way of thinking, consisting of directions and rules, structured in a systematic way in development activities with corresponding development products.” *Method engineering* deals with the engineering-based and systematic construction of methods as well as with their comparison and integration. Thus, methods themselves are objects of development. Brinkkemper (1996) defines: „Method Engineering is the engineering discipline to design, construct and adapt methods, techniques and tools for the development of information systems.”

According to the method engineering approach of the University of St. Gallen, a method is the “systematic and structured process of the development, modification and adaptation of software development methods through the description of method components and their relationships” (Heym 1993, translated). In this perspective, the components of a method are of central importance: Heym specifies the components ‘metamodel’, result, activity, technique and role.

A comparable approach is the one of Karlsson (2002). He represents his definition also as a UML diagram (fig. 1): “A method is normative prescribed actions performed by human actors in order to reach the actors’, or a subset of the actors’, goals”. A method aims at creating a product, e.g. models, as the final result or an intermediate milestone in the development process. This is called artifact by Karlsson: “An artifact is either a final or intermediate work product that is produced and used by actors during a system engineering project.”

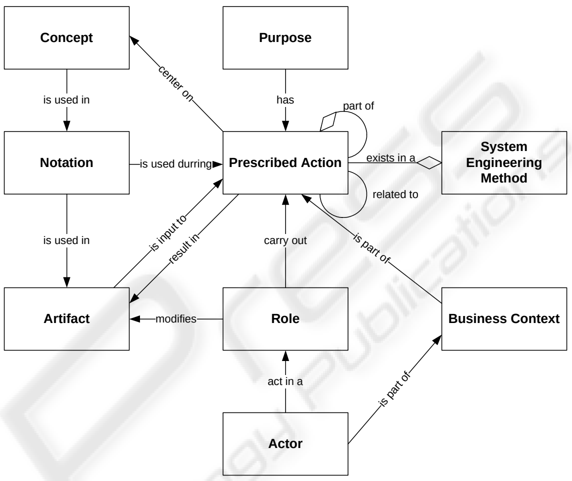

Figure 1: Method Metamodel (Karlsson 2002).

Despite the differences in the various approaches, many similar method components could be identified. Table 1 shows a comparison of the components in three selected approaches.

Table 1: Components of System Development Methods – Comparison (Goeken 2006).

	Metamodel	Technique	Result / Model	Role	Activities	Reason / Purpose	Concept	Notation	Principle	Tool
St. Gallen	X	X	X	X	X					
Brinkkemper		X	X		X			X	X	X
Karlsson	X	(X)	X	X		X	X	X		

We conclude that methods of systems development describe *who* (role) carries out *activities* in order to create a *product* (final or intermediate). The activities are supported by *techniques* and *tools*; the *product* (usually a document, a model or the final technical system) is created using a specific *notation*. The whole method is affected by *principles*. In a broader sense, Braun et al. (2005) characterize methods like follows: “a method is a process for systematically acquiring, representing and imparting knowledge.”

An (sub-) area of research, which might be of importance for IT governance related aspects, is the "situational method engineering". It treats the context-related adaptation and configuration of methods. The adaptation of methods in the area of system development happens in general via metamodeling and method fragments (Hofstede, Verhoef 1997; Brinkkemper et al. 1999; Saeki 1994). Brinkkemper et al (1999) build a framework for the classification of method fragments. It contains the three dimensions 'perspective', 'level of abstraction' and 'granularity'. The dimension perspective is divided into product and process, the abstraction into a conceptual and a technical level. Additionally, an ontology is proposed as an anchoring system. The classified fragments can be combined rule-based into methods.

It has to be discussed, if the notion and the understanding of methods can be used for IT management as well, or if methods of IT management are generally of different characteristics and thus, ideas of method engineering can not be transferred into this area. The notion of method, which currently is mainly affected by system development, possibly has to be extended in order to capture the rather management-driven tasks and processes of IT governance.

It seems that the components presented in fig. 1 and tab. 1 are sufficient generic. A method for IT management should be able to specify results and the techniques as well as the activities to reach them. Furthermore it should define responsibilities and roles for the defined activities. On the other hand, e.g. the component 'notation' is of particular importance in a method for system development. If this component is of the same importance in IT governance, and if the significance of the superior control in management-orientated methods has to be distinguished to a greater extent, remains to be seen.

Furthermore, the situational adaptation to a specific context, which is developed in "situational method engineering", seems to be relevant for IT governance. Having identified the "one-fits-all-"method as unreachable in the 1990s, also the different requirements and needs of companies in the IT management aspects can not be satisfied via one static method. Accordingly, an attribute-based approach to situational adaptation and configuration of the presented frameworks is desirable.

3 BEST PRACTICE FRAMEWORKS

3.1 Basics

As indicated, science offers little guidance to IT management and IT governance issues. Therefore, in the last ten years a range of open best practice frameworks (ITIL, COBIT) as well as proprietary frameworks were developed (Microsoft Operations Framework (MOF), IT-Service-Management (ITSM) of Hewlett-Packard, or the IBM IT Process Model (ITPM)).

These frameworks which are also subsumed under the developing topic "IT governance", describe goals, processes and organizational aspects of IT management and control.

One point regarding the development of best practice models is very interesting: practitioners from the business world consolidate their knowledge aiming to define generally accepted rules, processes, and characteristics. Despite the fact that a few scientists participate in the development of already mentioned frameworks such as COBIT, especially practitioners are members of the relevant committees and boards. (Johannsen, Goeken 2006, 2007)

From an academic viewpoint these best practice frameworks can be seen as an interesting object of research, not only because the models are widely spread but also because they incorporate a huge amount of consolidated knowledge. As mentioned before, a sound scientific discussion and foundation of these models is missing but could be fruitful.

3.2 COBIT

In the following we will focus on COBIT (Control Objectives for Information and Related Technology). COBIT describes a generic process model, that defines relevant processes and activities which should be found – according to the idea of best practice – in every well managed IT department or organization. Whereas earlier versions put the main focus on IT audits, the COBIT framework meanwhile developed to a full-blown support of IT management covering most relevant tasks and areas of this topic. (ITGI 2003, 2007)

In a macro-perspective the IT processes are arranged by grouping them into four so called control areas which are structured similar to the well known Deming cycle (plan, build, run, monitor) (figure 2)

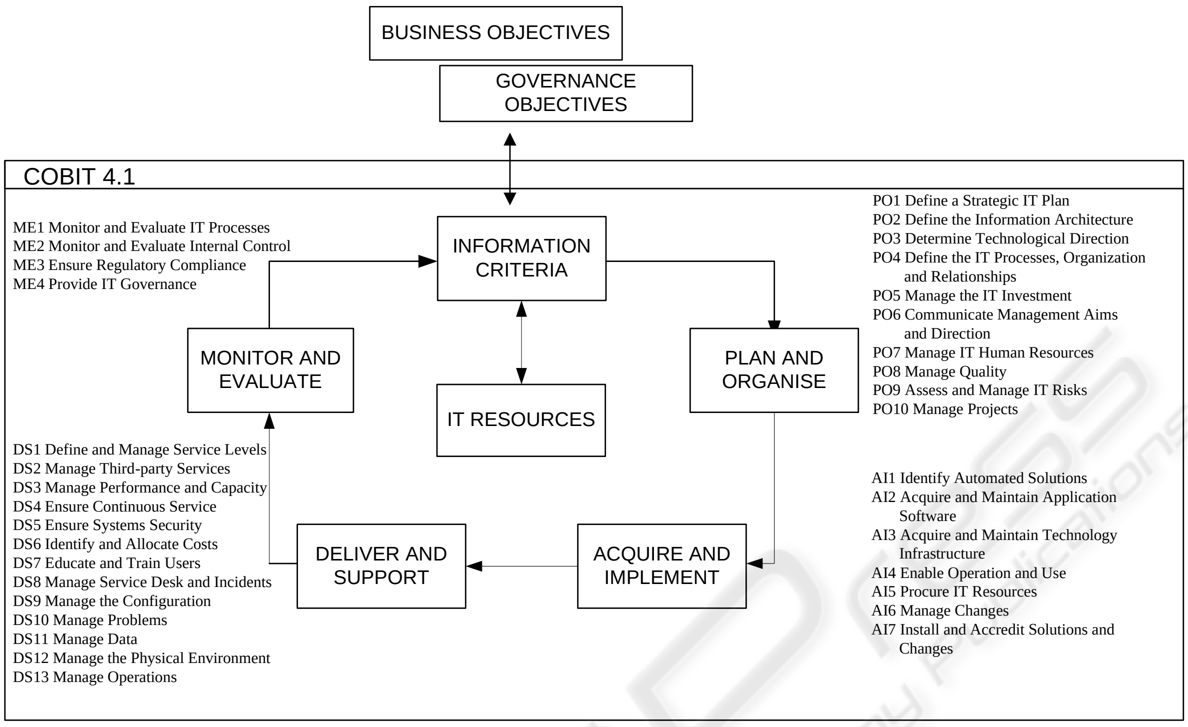
BUSINESS OBJECTIVES
GOVERNANCE OBJECTIVES
COBIT 4.1
INFORMATION CRITERIA
IT RESOURCES
MONITOR AND EVALUATE
PLAN AND ORGANISE
DELIVER AND SUPPORT
ACQUIRE AND IMPLEMENT
ME1 Monitor and Evaluate IT Processes
ME2 Monitor and Evaluate Internal Control
ME3 Ensure Regulatory Compliance
ME4 Provide IT Governance
PO1 Define a Strategic IT Plan
PO2 Define the Information Architecture
PO3 Determine Technological Direction
PO4 Define the IT Processes, Organization and Relationships
PO5 Manage the IT Investment
PO6 Communicate Management Aims and Direction
PO7 Manage IT Human Resources
PO8 Manage Quality
PO9 Assess and Manage IT Risks
PO10 Manage Projects
AI1 Identify Automated Solutions
AI2 Acquire and Maintain Application Software
AI3 Acquire and Maintain Technology Infrastructure
AI4 Enable Operation and Use
AI5 Procure IT Resources
AI6 Manage Changes
AI7 Install and Accredit Solutions and Changes
DS1 Define and Manage Service Levels
DS2 Manage Third-party Services
DS3 Manage Performance and Capacity
DS4 Ensure Continuous Service
DS5 Ensure Systems Security
DS6 Identify and Allocate Costs
DS7 Educate and Train Users
DS8 Manage Service Desk and Incidents
DS9 Manage the Configuration
DS10 Manage Problems
DS11 Manage Data
DS12 Manage the Physical Environment
DS13 Manage Operations

Figure 2: The COBIT Framework – Macro Perspective.

3.3 The Method Metamodel of COBIT

Beside the application area of COBIT we identified two further reasons to start metamodeling with COBIT.

- First, this framework is well structured in chapters and components, and therefore closed in itself and self-contained.
- Second, COBIT is comprehensive and covers (nearly) all tasks and processes an IT organization should carry out.

For example, ITIL (OGC 2007) is – like COBIT – comprehensive, but lacks of structure. On the other hand, e.g. CMMI (1999) focuses on a specific task (development), but has a coherent structure.

However, these existing structures primarily serve the purpose to present the framework consistently and structured. It supports the navigation and the usage of the framework but may not be mixed up with a metamodel. A goal of metamodeling the framework is to extract and present the underlying logical and semantically rich relationships.

Generally spoken, a metamodel is a model of a model. That means that initially there might be a model, which represents the real world or some part of it. The metamodel is the illustration of this model on the next higher level of abstraction. Here we use an abstraction mechanism which extracts the components of the underlying model. (This must be distinguished from the most common language abstracttion, which is used when the abstract syntax of a modeling language is represented in the metamodel).

We use the well known ER notation to represent our version of the COBIT metamodel. The analysis is stepwise and takes place in fragments which are in the end combined to one model. Initial point of the partial analysis is the entity type ‘*process’* and thus, it is also the later necessary entity which integrates the fragments.

3.3.1 Control Objectives, Activities and Results

In COBIT, the 34 IT processes produce *outputs* which vice versa are used as *inputs* in other processses. Input and output are *results*. According to this, the entity type result ‘is-a’ output or input of a process. Typical results on instance level are documents like reports on costs, risks or plans on IT strategy.

Moreover, a process consists of *control objectives*, which are statements of desired results or purposes to be achieved by implementing control procedures in a particular process. These control procedures should provide ‘reasonable assurance’, that business objectives will be achieved.

Furthermore, a process includes *activities*, which give a detailed description of what is done. These activities are carried out by specific persons like the CFO, the CIO, or an architect. Therefore, we link activities to the concept *role* (fig. 3).

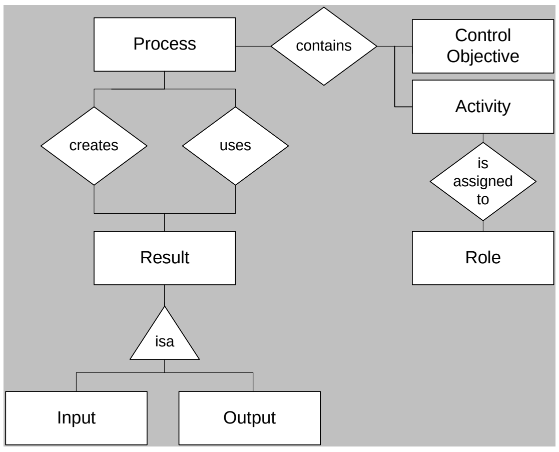

Figure 3: Control Objectives, Activities and Results.

3.3.2 Goals and Metrics

Each process of the framework has *goals*, which can be divided into business goals, IT goals, process goals and activity goals. As well these goals are in relationship with each other.

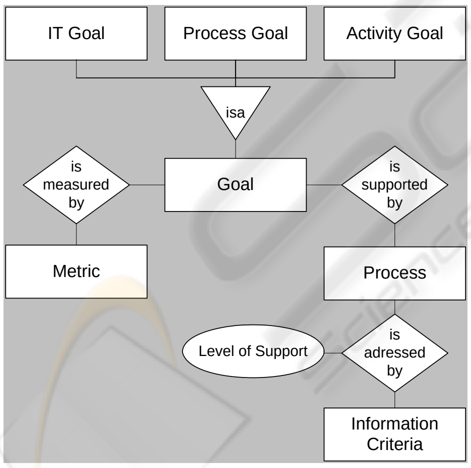

Figure 4: Goals and Metrics.

Thus, IT goals activate process goals, which in turn end up in activity goals (e.g. IT goals define what the business expects from IT; Process goals define what the IT process must deliver to support IT's objectives etc.). Each goal is measured with the aid of different *metrics*.

Furthermore, a process contains *information criteria*, which are abstract business goals. The information criteria proposed by COBIT are effectiveness, efficiency, confidentiality, availability, compliance and reliability. For every process COBIT indicates, if these criteria are supported. It is distinguished between a primary and a secondary relationship. Goals as well as metrics usually are neither considered as components in the usual method descriptions. Both are therefore candidates for the advancement of the metamodel (fig 4).

3.3.3 Maturity Model, IT Resource, Domain

Each process is assigned to one of four *domains*, which are arranged according to the life cycle (see above, fig. 2). Further components of COBIT are a *maturity model*, four domains and IT resources. Each process can be assessed by a maturity model to determine its level of maturation. This is the starting point for a continuous improvement of the process maturity and its controls.

In order to achieve results, a process needs the entity type *IT resource*. Implicit components as the life cycle orientation of COBIT could enter the metamodel as *principles*. This component is introduced by Brinkkemper, who mentions the "way of thinking". However, a principle can not be dedicated to a single entity type. Implicit basic principles form the framework as a whole and thus have to be put in another level of the metamodel.

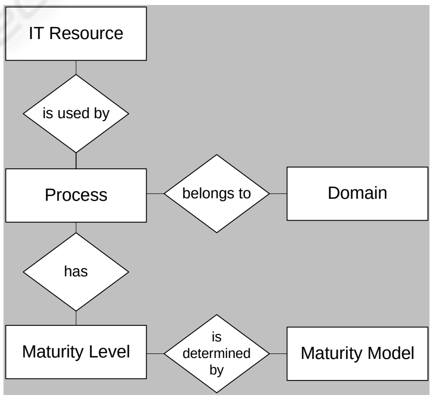

Figure 5: Maturity Model, IT-Resource and Domain.

3.3.4 IT Governance Focus Areas

Finally, each process has the attributes *process code* and *description*. The process code is a unique identifier of the process (e.g. PO5, see fig. 2). It consists of the abbreviation of the domain and a current number. Furthermore, each process supports a specific IT governance focus area. These *IT governance focus areas* "describe the topics that executive management needs to address to govern IT within their enterprises" (ITGI 2007).

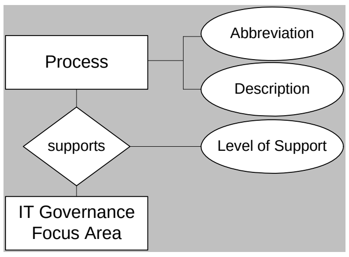

Figure 6: IT Governance Focus Areas.

For each process there is an indication if it addresses the focus area. Like above it is distinguished between a primary and a secondary relationship.

Figure 8 shows the integrated metamodel. The entity type process is used for the integration of the partial models presented above.

To conclude, by building the metamodel of COBIT, a lot of components could be identified, which can be found in the method metamodel presented in section 2 as well.

A method for IT governance also will define certain actions (processes) which generate specific results. In order to organize work and to assign responsibilities, it is necessary to relate roles to processes or activities.

In contrast to a development method, the assignment of roles to results often is of central importance, in order to represent more complex responsibility structures and decision rights. Therefore, the relationship 'assigned to', which connects 'Role' and 'Result', should be considered as a further, governance related extension.

Important components, which are not covered by systems development methods, are goals and metrics. Both are of central importance in governance frameworks.

In the following we will present some applications which demonstrate various benefits of the method metamodel.

4 APPLICATION AND USAGE OF THE METAMODEL

Several advantages accrue from representing IT governance frameworks like COBIT, ITIL or CMMI as method metamodels. In this section we will discuss some of the benefits and possible applications.

First, the representation allows the comparison of different frameworks on an abstract level. Once the components are extracted, frameworks can be examined and analyzed. Thus, other frameworks can be checked for completeness with the aid of the metamodel. Accordingly, one can deduce that ITIL - in contrast to COBIT - does not provide metrics and

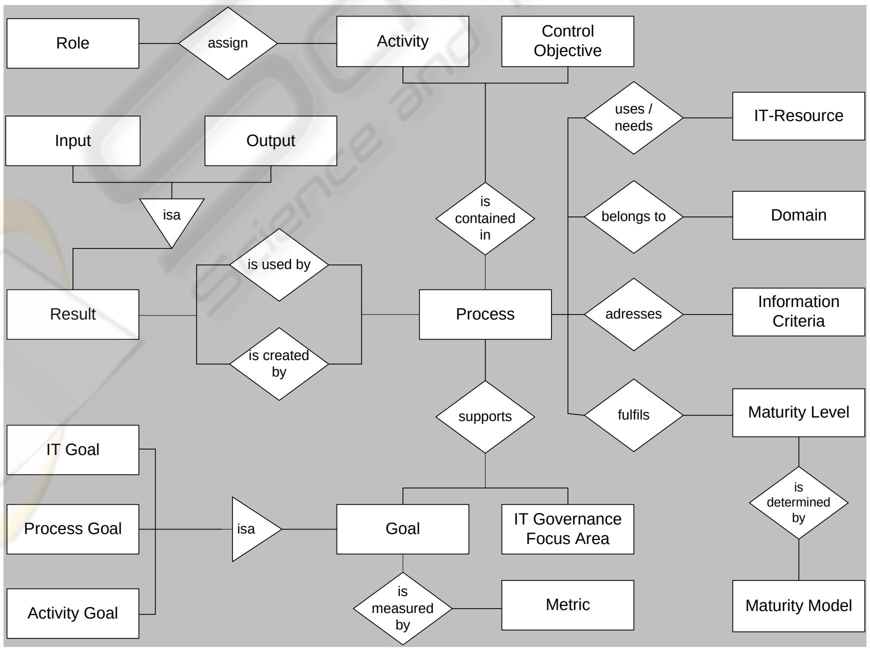

Figure 7: The Method Metamodel of COBIT.

other components for assessment to the extent COBIT does.

Another benefit of the metamodel is the integration of new or existing processes in the COBIT framework. This becomes apparent in the following example: Given the vast extend to which outsourcing is being practiced in today's IT departments it is an underrepresented issue in COBIT. With the aid of the metamodel a 'Control of the outsourcing'-process can be developed under guidance. In order to develop this process, the metamodel has to be instantiated.

In addition, the integration of the process into other existing IT processes can for example occur by linking the results. When inputs flow to the process and the output is used elsewhere, the new process becomes an integrated part of the overall IT process landscape.
One step further, a metamodel could be used to describe and analyze the linkage of various frameworks like COBIT, ITIL and CMMI with the aim of integrating them.

Besides, the metamodel can be the starting point for the representation of COBIT in an application system. The components and the logical and semantic relationships are necessary, e.g. for the implementation in a semantic network. We are currently developing a framework representation with this technology which allows the flexible navigation within the framework structures and the implementation of various views over the components (see fig. 9).

With a tool like this, we hope to support the implementation of governance frameworks in practice significantly.

5 CONCLUSIONS AND FUTURE RESEARCH

From our point of view, it is possible and fruitful, to interpret IT governance frameworks as methods. IT governance methods can learn from method engineering with respect to the engineering-based and systematic approach the latter follows.

In the article, we extracted the relevant components performing some kind of 'framework re-engineering' on COBIT. The resulting metamodel brings some benefits for comparing and integrating different frameworks. Furthermore, frameworks can be checked for completeness against the model. An interesting application might be the representation of IT governance frameworks in a tool which was

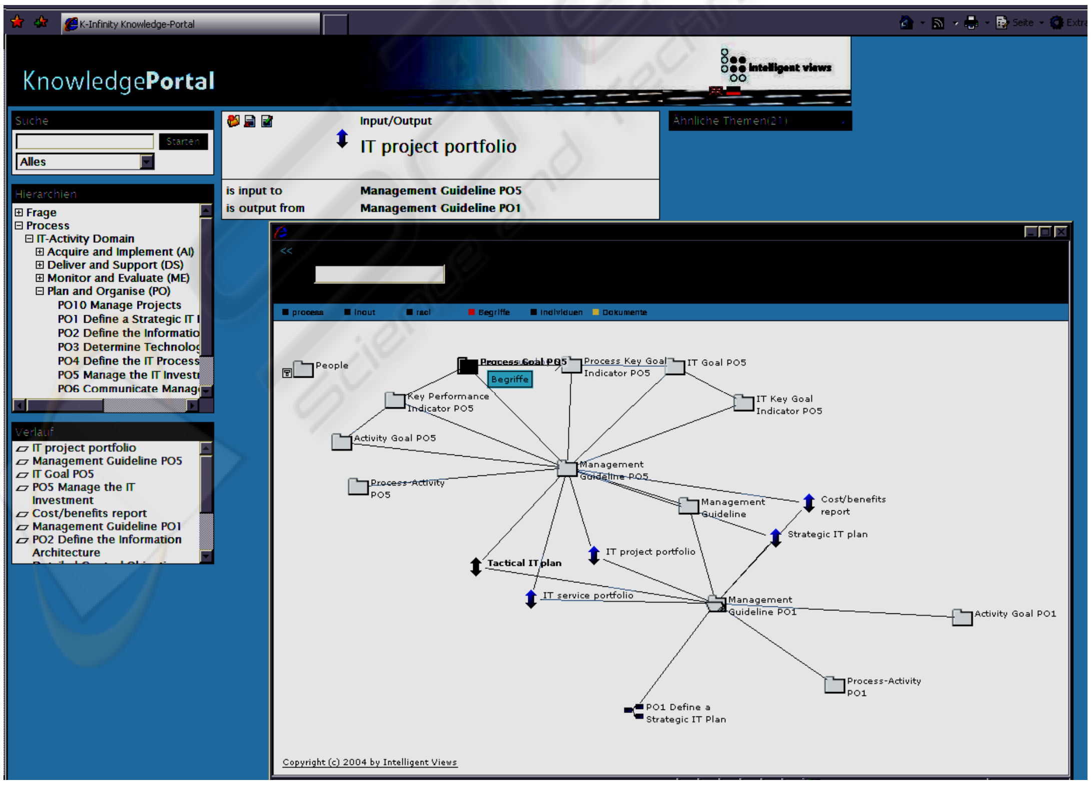

Figure 8: Representation of COBIT as a Semantic Net.

demonstrated in the previous section.

To give a widespread and holistic support for IT governance, it is not sufficient to metamodel one framework like COBIT. Instead, it is necessary to complement it with the knowledge of other frameworks and the findings of academic research (Steenbergen et al. 2007, Weill, Ross 2004). Therefore, the mapping of COBIT can only be the first step in the process of building a wider metamodel covering more than one framework and, therefore, covering various tasks and challenges of IT governance.

Another interesting area of development is the situation specific and/or enterprise specific adoption and configuration of governance models. We propose, that the mentioned approach of „situational method engineering" might be applicable to them as well. Because frameworks like COBIT and ITIL are seldom implemented completely and without modification, a methodological support for model adoption and configuration would be useful.

REFERENCES

Avison, D., Jones, J., Powell, P., Wilson, D., 2004.. Using and Validating the Strategic Alignment Model. In Journal of Strategic Information Systems, 13 (3), pp. 223-246.

Booth, Marilyn, E., Philip, G., 2005: Information Systems Management: Role of planning, alignment and managerial responsibilities. In Behaviour and information Technology vol.24 no.5.

Braun, C., Wortmann, F., Hafner, F., Winter, R., 2005. Method construction - a core approach to organizational engineering. SAC 2005: pp. 1295-1299.

Brinkkemper, S., Saeki, M., Harmsen, F.,1999. Metamodeling based assembly techniques for situational method engineering. In Information Systems 24 (1999) 3, pp. 209-228.

Brinkkemper, S., 1996. Method engineering: engineering of information systems development methods and tools. In Information and Software Technology 38 (1996), pp. 275-280.

Chan, Y. E., Huff, S. L., Barclay, D. W., Copeland, D. G., 1997. Business Strategy Orientation, Information Systems Orientation and Strategic Alignment. In Information Systems Research 8.

CMMI 1999 The Evolution of Process Improvement. Dezember 1999. Under:http://www.sei.cmu.edu/ newsatsei/features/1999/december/Background.dec99. htm, at July 4th 2003.

Goeken, M. 2006. Entwicklung von Data-Warehouse-Systemen. Anforderungsmanagement, Modellierung, Implementierung. Wiesbaden 2006. [in german]

Heym, M. 1993. Methoden-Engineering - Spezifikation und Integration von Entwicklungsmethoden für Informationssysteme. Dissertation, Institut für Wirtschaftsinformatik, Universität St. Gallen, St. Gallen. Hallstadt 1993.

Hevner, A. R., March, S. T., Park, J., Ram, S., 2004. Design Science in IS Research. In MIS-Quarterly 28 (2004) 1, pp.. 75-105.

Hofstede, A. H. M. T., Verhoef, T. F. 1997. On the feasibility of situational method engineering. In Information Systems 22 (1997) 6-7, pp. 401-422

ITGI 2003, IT Governance Institute. COBIT Implementation Guide, 2003, o.O.

ITGI 2006, IT Governance Institute. IT Governance Global Status Report, 2006. Under: www.isaca.org, at March 3rd of 2007.

ITGI 2007, IT Governance Institute. COBIT 4.1, 2007, o.O.

Johannsen, W., Goeken, M., 2006. IT-Governance - neue Aufgaben des IT-Managements, In HMD - Praxis Wirtschaftsinformatik, 250, 2006,pp. 7-20. [in german]

Johannsen, W., Goeken, M., 2007. Referenzmodelle für IT-Governance. dpunkt.verlag, Heidelberg. [in german]

Karlsson, F., 2002. Meta-Method for Method Configuration - A Rational Unified Process Case. Promotion, Linköping University, Faculty of Arts and Sciences, Thesis 61.

KPMG 2004. Summary of KPMG IS Governance Survey. KPMG LLP, London, September 2004.

OGC, Office of Government Commerce, 2007. ITIL V3. Service Strategy, London.

OGC, Office of Government Commerce, 2007. ITIL V3. Service Design, London.

OGC, Office of Government Commerce, 2007. ITIL V3. Service Transition, London.

OGC, Office of Government Commerce, 2007. ITIL V3. Service Operation, London.

OGC, Office of Government Commerce, 2007. ITIL V3. Continual Service Improvement (CSI), London.

Ralyté, J.; Brinkkemper, S.; Henderson-Sellers, B. (Ed.), 2007. Situational Method Engineering: Fundamentals and Experiences: Proceedings of the IFIP WG 8.1 Working Conference, 12-14 September 2007, Geneva, Switzerland, Heidelberg, 2007.

Reich, B. H., Benbasat, I., 1996. Measuring the linkage between business and information technology objectives. In MIS Quarterly, 20, 1, 1996, 55. MIS-Quarterly 28 (2004) 1, pp. 75-105.

Saeki, M.1994. Software Specification & Design Methods and Method Engineering. In International Journal of Software Engineering and Knowledge Engineering, 1994. http://citeseer.ist.psu.edu/490433.html.

Steenbergen, M., Berg, B., Brinkkemper, S., 2007. An Instrument for the Development of the Enterprise Architecture Practice. In Proceedings of the ICEIS 2007, Funchal, Portugal, 12-16. June 2007, pp. 14-22.

Weill, P, Ross, J.W.2004. IT Governance. Harvard Business School Press, Boston 2004.