Towards Conceptual Metamodelling of IT Governance Frameworks

Approach - Use – Benefits

Matthias Goeken
Frankfurt School of Finance & Management
m.goeken@frankfurt-school.de

Stefanie Alter
Frankfurt School of Finance & Management
s.alter@frankfurt-school.de

Abstract

Up to now, there has been little academic support for the challenges of IT governance/IT management. As a reaction, various best practice frameworks - like COBIT or CMMI - were developed. Due to their origin, these frameworks lack of a sound basis or scientific foundation. To undertake a step in this direction, we propose the use of modeling notation and techniques to represent frameworks as conceptual metamodels. Accordingly, we present the well-known framework COBIT metamodeled in a conceptual way and, thereby, represent the underlying logically and semantically rich structures. We furthermore discuss the benefits of using conceptual metamodeling to analyze frameworks coming from practice. Using the metamodel, we are also able to demonstrate ways to improve the frameworks and configure them according to the specific needs of an enterprise or an industry. To have a sound basis for any improvement we discuss the process of metamodelling and derive some requirements for “good” metamodelling.

1. Introduction

While there is little guidance for the management of IT in general and for specific challenges like business/IT alignment or risk management in particular [1, 5], the majority of computer science research deals with system development and related issues. To offer guidance for management tasks and governance challenges seems critical because in enterprises, usually a higher percentage of expenditure is spent on 'running IT' rather than systems engineering and development of new systems.

Due to the fact that there is a clear need for methodological support for current tasks and challenges of IT management and IT governance, it is surprising, that little attention is paid to these questions. [8] and [34] 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 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) that developed frameworks (e.g. COBIT and ITIL) to support management and governance of IT. These frameworks are well established in practice [16, 24]. However, there is a lack of theoretical foundation, from an academic viewpoint.

This paper undertakes steps towards the theoretical foundation of best practice frameworks by proposing to model them as conceptual metamodels. We will show that this theoretical foundation gives rise to various benefits and improvements also useful for practitioners. After that we introduce some best-practice frameworks especially the COBIT framework. In section 3 we discuss IT governance framework COBIT as conceptual metamodels. Therefore, we discuss the concepts 'model' and 'metamodel' as well as their relationship, and take a look on how metamodels need to be understood from a linguistic and an ontological point of view. Hereafter we present a metamodel of COBIT, the popular IT governance framework of the ISACA. Further, we discuss the advantages and the application of this metamodel (section 4) and show some research in progress.

2. IT Governance Frameworks

As mentioned in the introduction, science offers little guidance to IT management and IT governance issues. Therefore, in the last ten years a range of open best practice models (IT Infrastructure Library (ITIL) [31], Control Objectives for IT and related technology (COBIT) [17], Capability Maturity Model Integration (CMMI) [41]) as well as proprietary models were developed (Microsoft Operations Framework, IT-Service-Management of Hewlett-Packard, or the IBM IT Process Model).
These best practice models which are also subsumed under the developing topic “IT governance” describe goals, processes and organizational aspects of IT management and control [14, 18]. They are created in practice and are given to use in practice. 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 scientists also participate in the development of already mentioned frameworks such as COBIT or CMMI, especially practitioners are members of the relevant committees and boards.

From an academic point of view, these best practice models can be seen as an interesting object of research, not only because the models are widely spread in practice 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.

The Capability Maturity Model Integration (CMMI) published from the Software Engineering Institute (SEI) is a process improvement approach that provides organizations with the essential elements of effective processes. It can be used to guide process improvement across a project, a division, or an entire organization. CMMI helps integrate traditionally separate organizational functions, set process improvement goals and priorities, provide guidance for quality processes, and provide a point of reference for appraising current processes [22, 41].

In the following we focus on COBIT (Control Objectives for Information and Related Technology) [17]. Mainly, there were two reasons to start with the conceptual metamodeling of the COBIT framework.

First of all, this framework is well structured in domains, processes and other components and, therefore, closed in itself and self-contained. Secondly, COBIT is holistic and represents (nearly) all tasks and processes an IT organization should carry out. For example, ITIL is – like COBIT – holistic, but has a lack of structure. On the other hand, e.g. CMMI [41] focuses on a specific task (i.e. development), but has a coherent structure.

COBIT describes a generic process model, that defines relevant processes and activities which one should find – according to the idea of best practice – in an 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.

In a macro-perspective the IT processes are arranged by grouping them into four so called domains, which are structured following the well known Deming/life cycle (Plan, Do, Check, Act). For each of the 34 IT processes various components, such as business requirements, IT goals, controls and metrics as well as activities, resources, responsibilities and so on are defined. Furthermore, there are other components and specifications defined for each process, e.g. the roles that should be informed, should be consulted and so on. The “deep structure” of COBIT will be discussed in more detail in chapter 3, especially in section 3.3.

3. IT Governance Frameworks as Conceptual Models

3.1 Model and Metamodel

In IS research we use models as design artifacts [15] to abstract from reality and real world objects. The representation as a model is usually the first step of developing an application or software system, because, according to Lehman, “any program is a model of a model within a theory of a model of an abstraction of some portion of the world or of some universe of discourse” [27]. Our purpose is to strengthen the theoretical foundation by applying modeling to IT governance frameworks.

If the objects of research are models, and not the real world or the universe of discourse (UoD), we create models of models. Usually a “model of a model”, which is a higher level abstraction, is called metamodel. Going from the instance level (real world, UoD), consisting of instances (M0) to the model level (M1) and further to the metamodel level (M2) means the application of abstraction mechanisms (fig. 1).

[44] takes a systematic look at how these hierarchies are constructed and coins the term “metaization principle” for the operation that can repeatedly be applied. She points out, that there are different ways to come from real world instances to models and also from models to metamodels and so forth (see also [25]).

The metaization principle defines the primary abstraction mechanism for structuring the objects of the lower level. In other words, it specifies the relevant building blocks (phenomenon) the world consists of according the principle or viewpoint.

The metaization principle most often used in information systems is the “linguistic metamodeling”: For example, when using Chen’s E/R-Model-Syntax (M2) to represent a part of the real world (M0), on model level (M1) we can only use entities, relationships and attributes as relevant building blocks. Therefore, the level Mx makes us to structure the objects on Mx-2 in accordance to its predefined
building blocks (e.g. an ‘order’ can only be an entity or a relationship (relating ‘product’ and ‘customer’) or an attribute (which is unlikely)).

The representation of the object is matter of the so-called concrete syntax which defines the assignment of symbols to a representation. The representation, e.g. the shape of the symbols used, is called notation. A modeling language usually has one abstract syntax, which is the result of linguistic metaiziation, but may have several notations (e.g. see the manifold of E/R-model notations).

![Fig. 1 Ontological and linguistic metamodeling](image)

It is necessary to stress that linguistic metamodeling is not the only way to perform metamodeling, because various mechanisms of abstraction can be used for different purposes and applications. [20, 21] e. g. emphasis the ‘ontological metamodeling’. Other researchers also work on ‘physical metamodeling’ [2, 10]. For our purpose, the ontological metamodeling is of primary importance. In contrast to linguistic metamodeling, the ontological metamodels deal with the classification of model elements according to their content [2, 3]. By ontological metamodeling we define metatypes on M_X, which describe what concepts exist on M_X as well as their properties. Applying this kind of metamodeling might result in denoting M_0 object ‘order’ as a ‘sales object’ on M_2, and ‘customer’ as ‘business partner’ (together with supplier).

The differences between ontological and linguistic metamodeling are illustrated in figure 1, which also illustrates our approach. On M_0 there is a concrete activity within an IT process (a specific help desk process, ticket # 2009-42). The activities performed are instances of the COBIT activities “detect and record incidents” within “DS 08”. In COBIT the definition of processes and their activities are in natural language but semi structured in a form-like manner (in contrast, ITIL processes are described more narrative (prose)), but it contains some process diagrams. Referring to the ontological metailization, “detect and record incidents” can be classified as an “activity” on metamodel level. This ontological metaization from M_1 (where COBIT resides) to the meta level M_2 (where our model resides) is the nucleus of our approach and the modeling activities, described in the following chapter. In addition to the identification of components like “activity” we further model the relationships between the identified components. This, for example, reveals, that an activity is always related to a role.

In order to model some portion of the world (which might be a model), one needs a language as well as a method with procedures, which supports the identification and representation of relevant objects. The language is often considered as the “way of modeling”, the procedures as the “way of working” [45]. In information systems research, the emphasis is usually on the way of modeling, the language/notation, and not much attention is directed towards the problem of filling the models, that is, instantiating the model with the knowledge of the UoD. Brinkkemper [7] adds the “way of thinking”, which refers to directions, rules, principles and guidelines, that are used in the modeling process. In the following, we focus on the way of thinking by discussing guidelines and principles for metamodeling. From our point of view, this is a prerequisite for defining ways of working. The latter is not addressed in this paper, but it is an area of future research.

As metamodels represent the “deep structures” of a model, the language (way of modeling) used to formulate the model must be able to represent the concepts of the model in an appropriate way. Here, a dialect of the E/R-approach, the extended E/R-model, eERM will be used [36]. This means, we focus on the static aspects of the framework and are not able to capture e.g. information flows directly. In fig. 1 the language is depicted because our model is a linguistic instantiation of the M3 Metamodel.
3.2 Guidelines of Metamodeling

In computer science and information systems various modeling languages have been developed but only in the recent past the researchers discuss how the domain knowledge could be transferred into the models. To fill this lack some researchers propose requirements engineering and a kind of stepwise method of metamodeling [26], others believe that describing a metamodel in an epistemological way has to be the first step in any research activity including a metamodel [38]. We agree both and believe further that a high-quality metamodel is a sound basis for the following steps in research [30]. Hence, we derive some guidelines of ‘good’ metamodeling in the following [37].

Because the metamodel is defined as ‘a model of a model’ the so called guidelines of modeling probably can be applied on metamodels as well. [37, 38] propose six principles to raise the quality of information modeling. These principles are:

1. Principle of construction adequacy
2. Principle of language adequacy
3. Principle of economic efficiency
4. Principle of clarity
5. Principle of systematic design
6. Principle of comparability

Principle of construction adequacy means a consensus according to the represented problem and about the type of construction. But it does not cover the interrelation between models of different types. This principle requires a pragmatic way thinking and modeling and a purpose orientation. The construction of a model should be adequate to problem and purpose.

Principle of language adequacy means that the chosen language fits the purpose of the model. For our research we choose the eERM to metamodel COBIT. We use the metamodel to compare, map and integrate different frameworks because of their components but not their behavior or dynamic structure. Therefore, eERM is an adequate language because we would not model behavioral aspects of COBIT. This principle also includes completeness and consistency between the model and the metamodel of the chosen language. That means that a model does not include any symbols or items which are not specified in the metamodel. In the context of metamodeling we probably need an extension of the eERM notation according to integration and mapping of frameworks.

The principle of economic efficiency formulates the economic restrictions every activity in an economic institution is exposed to. There is no need for changes or extensions in a metamodeling approach.

The principles of clarity and systematic design deals with the comprehensibility of the model design. Systematic design requires inter-model consistency. These principles are important for the integration of frameworks by using metamodels. For instance, the principle of systematic design demands a comprehensive metamodel that includes all components of the frameworks, which should be integrated.

In addition the principle of comparability is one of the major principles in a metamodel environment. Metamodels are often used to compare and integrate models on an abstract level. Therefore, comparability is a critical point if metamodels are focused.

With some restrictions these six principles could also be used for metamodeling. One reservation is that the guidelines of modeling often use the comparison with the reality. However dealing with metamodels implies that models are the basis of modeling. So it is unfeasible to validate the semantics of the metamodel against the reality. But within these restrictions it could be a first conclusion that the guidelines of modeling could also be used for metamodeling, because the metamodel is also ‘just’ a model.

In our opinion there has to be an extension to these six principles in order to use them for metamodeling purposes.

To come from model to metamodel the modeler chooses a way of abstraction (see section 3.1). This principle of metaization, for instance the ontological or the linguistic metaization, has intense consequences for the metamodel (see the following sections). Especially the ontological metamodeling which we use in our metamodel requires many decisions from the modeler. The user of the metamodel needs the information, which principle of metaization the modeler has used for the metamodel. As a result the first guideline of metamodeling should be: A metamodel has to reveal its principle of metaization.

We will use our metamodel for the combination and integration of different frameworks. Due to this field of application, the different users and the high level of abstraction the metamodel needs a clear mapping between the ‘universe of discourse’ and the words and symbols which will describe it ([9], [33]). That means that the linguistic defects like synonyms and homonyms are minimized or removed. The following example shows the relevance for metamodeling: Many IT governance frameworks are using the component goal. But each framework got its own definition of what goal means. Here we see items with the same name but with different meanings. But if we compare the generic goals of the CMMI standard with the control objectives of the COBIT standard we will find out that they have got almost the same meaning but a completely different concept. The second guideline for metamodeling should be the clear
mapping between a concept and its meaning in the scope of the metamodel.

By metamodeling existing frameworks the difficulty is that the differences between notion and connotation are part of the model. In the following we present at first a descriptive metamodel of the COBIT framework. After that we will discuss the effect of the differences between notion and connotations on the possible integration with other frameworks. The importance of this guideline increases when the metamodel is used by different users or different models should be integrated.

As guideline three we propose the use of semantically rich connections (the concept depends on the chosen language; in this paper we use eERM- that means semantically rich relationship types). Due to the fact, that the model (UoD) we (meta-)model in the following is not all over consistent, clear and systematically designed we often had to choose among several relationships between two entities. It is evident that a relationship type like ‘is created by’ or ‘contains’ includes more information than a relationship type with a conjunction of ‘have’ or with other elementary relationship types (For instance the relationship type ‘P-MM’ between the entities process area and maturity model in fig 3.). But to differentiate between two semantically richer relationships we often need further information.

To summarize this section we could say that the well known guidelines of modeling could apply on metamodels if we pay attention to some restrictions. Besides these six principles a metamodel should observe the following guidelines:

- A metamodel reveals its principles of metaization.
- A metamodel has an unambiguous mapping between the universe of discourse and the words and symbols which name and describe it.
- A metamodel has semantically rich connections.

### 3.3 Metamodelling the Best Practice Framework COBIT

In this chapter we present a metamodel of COBIT, which is developed in more detail in [12, 13] and show how frameworks of IT governance can be modeled considering the aforementioned principles and guidelines. Furthermore we extract the main benefits of metamodeling. Mainly, there were two reasons to start with the descriptive metamodeling of the COBIT framework. As mentioned before COBIT is well structured in processes and other repetitive components and, therefore, closed in itself and self-contained. Secondly, COBIT is holistic and represents (nearly) all tasks and processes an IT organization should carry out. However, these existing structures primarily serve the purpose to present the framework consistently and structured. The COBIT manual supports the navigation and the usage of the framework but may not be mixed up with a metamodel. A decisive difference between the manual and a metamodel is the goal of metamodeling to extract and present the underlying logical and semantically rich relationships (see guideline 3).

In COBIT 34 IT processes are presented which produce one or more outputs which vice versa are used as inputs in other processes. Input and output are results. According to this, the entity type result ‘isa’ output or input of a process (see fig. 2). 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 (see also section 4).

Each process of the framework has goals, which can be divided into business goals, IT goals, process goals and activity goals. The goals again are in relationship with each other. 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 and so on). Each goal is measured with the aid of different metrics (key goal indicators and key performance indicators). 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 states 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 method descriptions [6] nor in the widespread modeling notations like EPC ((Event-driven Process Chain) or BPMN (Business Process Modeling Notation) [23 extended performance measures into BMPN].

From an IT governance point of view, goals and metrics are of high importance because in order to control, govern and manage, you have to quantify the relevant facts (“You cannot manage without measuring”). Each process is assigned to one of four domains, which are arranged according to the life cycle. Further components of COBIT are a maturity
model, 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 process improvement of the process maturity and its controls. In order to achieve any results, a process needs the entity type IT resource.

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 [17]. 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. Implicit components as the life cycle orientation of COBIT could enter the metamodel as principles. 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.

Finally, each process has the attributes process code and process description. The process code is a unique identifier of the process. It consists of the abbreviation of the domain and a number. Figure 2 shows our ontological metamodel of COBIT. (For further information to the model see [12, 13])

In the following we present and discuss how our approach fulfills the principles of modeling and the proposed guidelines.

To fulfill the principle of construction adequacy (1) we had to find a consensus according to the represented problem and about the type of construction. The construction of our metamodel should be adequate to problem and purpose. Firstly we will use the metamodel to clarify the components of IT governance frameworks and after that we use the metamodels to compare and integrate different frameworks. We find a consensus that the represented problems are the components of IT governance frameworks and their underlying structure. That is the reason why we use conceptual metamodels as the type of construction.

To fulfill the principle of language adequacy (2) for our metamodel, we use the well known extended E/R notation (eERM) to represent our version of the COBIT metamodel. We only used elements which are part of this notation and structured the model in a clear and schematic way. The chosen language fits with the purpose of our metamodel. For instance eERM – like our metamodel – has no dynamic components. The principle also includes completeness and consistency between the model and the metamodel of the chosen language.

![Fig. 2 Ontological metamodel of COBIT](image-url)
We fulfill this principle because we do not include any symbol or items which are not specified in the eERM metamodel. But for using our metamodel in the context of mapping and integration we might need an extension of the eERM notation.

The principles (4) und (5) claim clarity and systematic design. We conform to them by using a clear structural guideline for arranging the components of the metamodel. Furthermore, we regulated the layout.

The principle of comparability (6) will be critical if it is applied to metamodels which are used for mapping and integration of different IT governance frameworks (see section 4).

The proposed new guidelines of metamodeling are fulfilled in the following way:

Guideline 1: A metamodel reveals its principles of metaization.

Our metamodel is an ontological metamodel. The linguistic metamodel of COBIT would describe the design of the standard. A linguistic metamodel of COBIT would describe that each process is presented on four pages, including a RACI chart on the middle of page three which has twelve columns and so on. Ontological metamodelling demands for a socialization and an educational background within the domain of IT management. Therefore, it is important for the metamodel user to know which way of metaization was used. To follow guideline 1 we have to reveal the metaization principle we used. We reveal that our metamodel is an ontological one by naming the figure. There might be other ways to show the principle of metaization but to use the caption seems to be pragmatic and sufficient.

Guideline 2: A metamodel has an unambiguous mapping between the universe of discourse and the words and symbols which name and describe it.

We use the COBIT notion for our metamodel. The concepts of the model are used likewise for the metamodel. The user of the COBIT framework has some commitment to the COBIT “language”.

Guideline 3: A metamodel has semantically rich connections.

We did not use any simple relationship types to connect the entities. If we had the choice between two or more opportunities we use the semantically richest relationship type to describe the connection between the entities. We use the relationship type activity-role in figure 3 only to illustrate the differences.

Additionally we drop the component “management guideline” which, therefore, is not part of our metamodel. In the COBIT framework, the “management guideline” denotes page three of the IT process description, but in our opinion, is only a structural component and not semantically rich - which is claimed by guideline 3.

The principles of modeling and the proposed guidelines for metamodeling support and promote the quality of (meta) models. [29] describe further influencing factors of quality in conceptual models. The examination had shown that our metamodel fulfills the principles of modeling and the recommended extensions. In the following section we describe application and usage of our approach.

4. Applications and Usage of IT Governance Frameworks as Conceptual Metamodels

We assume that several advantages accrue from representing IT governance frameworks like COBIT, ITIL or CMMI as conceptual metamodels, and that metamodels can be a helpful support for analysis and further advancement in the research on IT governance frameworks. In the following we will discuss some of the resulting 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 other components for assessment to the extent COBIT does. Another benefit of this approach is that the quality of a metamodel is verifiable by the extent of fulfilling the guidelines of metamodeling. This metamodel might be a sound basis for an improvement of the framework. An analysis of the presented metamodel of COBIT may e.g. raise questions like:

- Why are activities related to a role, while control objectives are not assigned to a role or a person?
- Why are activities not measured by metrics?
- Why are activities and control objectives not directly related?

The following example shows one possibility to improve the framework. The items activity and control objective are components of the COBIT framework. Both are related to Process but are not related to one another. The control objectives have many substantial overlappings with activities which complicate using the framework. The overlapping between control Objectives and activities can be eliminated by dropping the relationship between activity and process as shown in figure 3.

Another example could be the component role. The unmodified COBIT framework (Fig. 3) could not support any complex structures of responsibilities. In reality it might be necessary that the responsibility for
an activity is strictly separated from its supervision or from another activity e.g. cause of a trade-off. Simonsson and Johnson [39, 40] present another possibility to use and improve the IT governance framework COBIT. They show that COBIT does not support most needs of previously identified concerns of literature and practitioners, because it lacks in providing information on how COBIT should be implemented. Their IT organization modeling and assessment tool (ITOMAT) firstly includes a modeling language which provides “a descriptive representation of how IT is governed within the assessed company” and secondly an analysis framework which “provides support for the evaluation of whether the given IT governance structure is good or bad.” [39, p. 3]

Another benefit of the metamodel is the integration of new or existing processes in the COBIT framework. The metamodel prescribes the relevant components which should be implemented in order to fulfill new tasks of IT management on a solid basis. This becomes apparent in the following example: The area outsourcing is hardly represented in the COBIT framework. However, outsourcing is an essential component of their IT strategy for some companies.

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 occur for example by linking the results of the new process to existing processes (input-output-relationships between processes). When inputs flow to the process and the output is used elsewhere, the new process becomes part of the overall IT process landscape. One step further could be the metamodel based fusion of frameworks like COBIT, ITIL and CMMI. This might be of importance if one framework covers aspects, which are missing in another one. E.g. the new developments of the SEI (CMMI for Services; CMMI for Acquisition) can be a complement for COBIT, as acquisition, like outsourcing, is not covered adequately by COBIT. A metamodel based integration will allow a closer fit and can guide the models amalgamation on lower level.

CMMI is commonly perceived as a maturity model. COBIT includes a maturity model as well. Eyeballing this component might be a point of departure for mapping the frameworks. But after taking a deeper look at the metamodels it becomes understandable, that the CMMI standard includes components like goals or procedures which are not a part of the maturity model of COBIT. This shows the difficulties caused by linguistic defects (homonyms, synonyms, antonyms) and underlines the advantages of integration by using metamodels.

To deal with this problem, we hope to gain from database research, namely from schema integration. Conrad notes that the “essential task in integrating heterogeneous data sources is the construction of a common and uniform description of the integrated data”. As schema integration is a method to develop a reconciled representation of multiple schemas, we hope to adopt essential ideas to the challenge of integrating governance frameworks while using their metamodels. The core problems in schema integration are [35]: schema matching, i.e. the identification of correspondences between schema objects, and schema merging, i.e. the creation of a unified schema based on the identified mappings. The first problem refers to the problem of identifying homonyms, synonyms, antonyms. The second is how to deal with the identified correspondences. Literature suggests various strategies [4, 28].

Another strategy for the discussed topic of governance frameworks might be the development of a generic metamodel, into which the frameworks like COBIT, ITIL or company specific models can be “plugged in”. [11] uses the schema integration taxonomy of Spaccapietra et al. ([42, 43 and 32]) which gain currency in database research. This taxonomy might be a possibility to clarify the conflicts between metamodels of different IT governance frameworks.

The findings from these researchers might be an enrichment of our research. An example is the need for keeping the integrated schema to a minimum. That means that redundant model fragments should not be part of the integrated schema again. In addition these findings fulfill the principle of economic efficiency, of clarity and of systematic design.

**Fig. 3 Control objectives, activity and role**

Another benefit of the metamodel is the integration of new or existing processes in the COBIT framework. The metamodel prescribes the relevant components which should be implemented in order to fulfill new tasks of IT management on a solid basis. This becomes apparent in the following example: The area outsourcing is hardly represented in the COBIT framework. However, outsourcing is an essential component of their IT strategy for some companies.

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 occur for example by linking the results of the new process to existing processes (input-output-relationships between processes). When inputs flow to the process and the output is used elsewhere, the new process becomes part of the overall IT process landscape. One step further could be the metamodel based fusion of frameworks like COBIT, ITIL and CMMI. This might be of importance if one framework covers aspects, which are missing in another one. E.g. the new developments of the SEI (CMMI for Services; CMMI for Acquisition) can be a complement for COBIT, as acquisition, like outsourcing, is not covered adequately by COBIT. A metamodel based integration will allow a closer fit and can guide the models amalgamation on lower level.

CMMI is commonly perceived as a maturity model. COBIT includes a maturity model as well. Eyeballing this component might be a point of departure for mapping the frameworks. But after taking a deeper look at the metamodels it becomes understandable, that the CMMI standard includes components like goals or procedures which are not a part of the maturity model of COBIT. This shows the difficulties caused by linguistic defects (homonyms, synonyms, antonyms) and underlines the advantages of integration by using metamodels.

To deal with this problem, we hope to gain from database research, namely from schema integration. Conrad notes that the “essential task in integrating heterogeneous data sources is the construction of a common and uniform description of the integrated data”. As schema integration is a method to develop a reconciled representation of multiple schemas, we hope to adopt essential ideas to the challenge of integrating governance frameworks while using their metamodels. The core problems in schema integration are [35]: schema matching, i.e. the identification of correspondences between schema objects, and schema merging, i.e. the creation of a unified schema based on the identified mappings. The first problem refers to the problem of identifying homonyms, synonyms, antonyms. The second is how to deal with the identified correspondences. Literature suggests various strategies [4, 28].

Another strategy for the discussed topic of governance frameworks might be the development of a generic metamodel, into which the frameworks like COBIT, ITIL or company specific models can be “plugged in”.

[11] uses the schema integration taxonomy of Spaccapietra et al. ([42, 43 and 32]) which gain currency in database research. This taxonomy might be a possibility to clarify the conflicts between metamodels of different IT governance frameworks.

The findings from these researchers might be an enrichment of our research. An example is the need for keeping the integrated schema to a minimum. That means that redundant model fragments should not be part of the integrated schema again. In addition these findings fulfill the principle of economic efficiency, of clarity and of systematic design.

**Fig. 3 Control objectives, activity and role**

Another benefit of the metamodel is the integration of new or existing processes in the COBIT framework. The metamodel prescribes the relevant components which should be implemented in order to fulfill new tasks of IT management on a solid basis. This becomes apparent in the following example: The area outsourcing is hardly represented in the COBIT framework. However, outsourcing is an essential component of their IT strategy for some companies.

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 occur for example by linking the results of the new process to existing processes (input-output-relationships between processes). When inputs flow to the process and the output is used elsewhere, the new process becomes part of the overall IT process landscape. One step further could be the metamodel based fusion of frameworks like COBIT, ITIL and CMMI. This might be of importance if one framework covers aspects, which are missing in another one. E.g. the new developments of the SEI (CMMI for Services; CMMI for Acquisition) can be a complement for COBIT, as acquisition, like outsourcing, is not covered adequately by COBIT. A metamodel based integration will allow a closer fit and can guide the models amalgamation on lower level.

CMMI is commonly perceived as a maturity model. COBIT includes a maturity model as well. Eyeballing this component might be a point of departure for mapping the frameworks. But after taking a deeper look at the metamodels it becomes understandable, that the CMMI standard includes components like goals or procedures which are not a part of the maturity model of COBIT. This shows the difficulties caused by linguistic defects (homonyms, synonyms, antonyms) and underlines the advantages of integration by using metamodels.

To deal with this problem, we hope to gain from database research, namely from schema integration. Conrad notes that the “essential task in integrating heterogeneous data sources is the construction of a common and uniform description of the integrated data”. As schema integration is a method to develop a reconciled representation of multiple schemas, we hope to adopt essential ideas to the challenge of integrating governance frameworks while using their metamodels. The core problems in schema integration are [35]: schema matching, i.e. the identification of correspondences between schema objects, and schema merging, i.e. the creation of a unified schema based on the identified mappings. The first problem refers to the problem of identifying homonyms, synonyms, antonyms. The second is how to deal with the identified correspondences. Literature suggests various strategies [4, 28].

Another strategy for the discussed topic of governance frameworks might be the development of a generic metamodel, into which the frameworks like COBIT, ITIL or company specific models can be “plugged in”.

[11] uses the schema integration taxonomy of Spaccapietra et al. ([42, 43 and 32]) which gain currency in database research. This taxonomy might be a possibility to clarify the conflicts between metamodels of different IT governance frameworks.

The findings from these researchers might be an enrichment of our research. An example is the need for keeping the integrated schema to a minimum. That means that redundant model fragments should not be part of the integrated schema again. In addition these findings fulfill the principle of economic efficiency, of clarity and of systematic design.
Another corresponding research could be the Viewpoint based Meta Model Engineering. [26] present an approach for method engineering applied on work systems engineering. They presented five steps to an integrated metamodel which is made up of different metamodel fragments. These fragments are viewpoint based and designed and validated by the requirements of each viewpoint. The relationships between the viewpoints are mapped in a ‘viewpoint relationship diagram’. To transfer these five steps to the integration of metamodels of IT governance frameworks might be a further step for a systematic combination of different best practice framework.

5. Conclusions and Future Research

In this paper we discussed and presented a way to represent the popular IT governance framework COBIT as a conceptual metamodel. From our point of view, it is possible and fruitful to interpret IT governance frameworks as metamodels. IT governance models can learn from a rigid formalization and a systematic approach. The intention was to demonstrate that metamodeling is a useful technique to gain a theoretical foundation on the one hand, and to analyze, compare, and integrate them on the other.

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.

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 framework structures and the implementation of various views over the components. In this respect, other approaches like [19] choose a process oriented viewpoint. Using the semantic networks, we prefer a knowledge oriented approach which might be a helpful contribution.

Another interesting area for further research could be the situation specific and enterprise specific adaptation and configuration of governance models, because frameworks like COBIT, ITIL or CMMI are seldom implemented completely and without modification.

Metamodeling can be the starting point for a methodological support for model adaptation. In further research, the metamodel presented should be made configurable by introducing and modeling variability on the instance as well as on the metamodel level.

6. References