Viewpoint-Based Modeling—Towards Defining the Viewpoint Concept and Implications for Supporting Modeling Tools

Klaus Fischer[1], Dima Panfilenko[1], Julian Krumeich[1], Marc Born[2], Philippe Desfray[3]

DFKI GmbH[1], Saarbrücken, Germany
ikv++ technologies ag[2], Berlin, Germany
SOFTEAM[3], Paris, France
{firstname.surname}@dfki.de, born@ikv.de, philippe.desfray@softeam.fr

Abstract: Viewpoint-based modeling is an important recent development in software engineering. It is likely to boost the wider use of modeling techniques because it allows to tailor existing tools with respect to the different stakeholders in software design. The paper reports on results from the VIBAM project in which viewpoint concepts are investigated. We give an overview of the most important contributions from literature regarding viewpoint concepts from which we derived the position that we take in the VIBAM project. After presenting VIBAM's position we derive features that we consider important for tools that support viewpoint features. We plan to integrate these features in the commercial modeling tools MODELIO and MEDINI ANALYZE to the end of the VIBAM project.

1 Introduction

In the course of constructing large-scale and complex systems, models are a prevalent means for gaining a better understanding of the underlying artifact that is to be engineered. Especially in traditional engineering disciplines [BKS10], graphical representations of models are widely used to design, describe and discuss various types of systems (take for instance building plans or engineering drawings). Apart from these traditional engineering disciplines, models are being increasingly used in the course of developing information systems as well as information systems architectures. In this regard, they do not only serve as a means for building systems from scratch, they are also used to capture current as-is states, e.g. for documentation purposes, with the eventual goal to derive optimized to-be states.

As a consequence, modeling techniques are commonly used nowadays—in particular standard modeling languages such as the Unified Modeling Language (UML) or the Business Process Modeling Notation (BPMN), but also proprietary and domain specific modeling languages such as those supported by Business Process Management (BPM) or Enterprise Architecture (EA) tools. However, according to our estimation, their usage has reached a significant threshold and a wider usage and diffusion needs more added value and perceived Return on Investment (ROI) by the end users. This can only be achieved if modeling techniques as well as corresponding tools put actual stakeholders more into the focus during the modeling process.

1.1 Motivation and Contribution of this Paper

According to Stachowiak [Sta73], models are a representation of a mental or physical object, in which this object can again be a model itself. Due to the reduction feature of models, they do not comprise every detail of the underlying object, since their purpose is to describe and analyze complex systems in a manageable way. However—even though models aim at reducing complexity—in large-scale systems, they are often still too complex, which is a common modeling problem [BW06]. This becomes even worse if only a single, rigid perspective is provided on models and moreover if models are used for cooperative and collaborative system design.

In such settings, many different stakeholders are working together to design a system. Each of them has very own skills, responsibilities, knowledge and expertise [FKN+92]; and thus, a very unique perspective on the system design. As a consequence, each person or group of people that is in charge with the system design would like to view and manipulate the models or model fragments according to their particular needs [DQvS08]. To foster the modeling usage considering these demands especially in the course of constructing large-scale systems, **Viewpoint-Based Modeling** (VBM) is an increasingly used technique to reduce complexity and adapt an overall model to stakeholder specific fragments in a successful manner [GBB12]. In doing so, the actual stakeholders are being put more into the focus of the modeling process, which results in a higher perceived value for them. By utilizing stakeholder-specific viewpoints on a model, its overall understanding and productivity will increase [FKN+92]. As a direct consequence, the viewpoint concept leads to better conceptual models, which is proven by several studies. For example, according to [EYA+05], applying the viewpoint concepts effectively helps to cope with the overall size of a given problem domain. Even though the usefulness of VBM is obvious and objectively proven, the concept's meaning is still quite fuzzy, since it is heavily overloaded in literature and frequently only defined in an informal manner [MP00], [GBB12].

Thus, this paper contributes towards knowledge on VBM in two respects: On the one hand, it provides a systematic clarification of the viewpoint concept on a general abstract level. In doing so, researchers will benefit from this definition and further research on VBM can be conducted on this basis. On the other hand, this paper provides various features that need to be considered in modeling tools in order to achieve a successful utilization of the viewpoint concept. In doing so, tool builders and vendors will benefit from this catalogue of features in order to validate their tools regarding VBM and to have a guideline for building new tools which support VBM.

1.2 Scientific Methodology and Organization of the Paper

The results of the paper originate from the European research project ViBaM that aims at investigating the viewpoint concept from a scientific perspective and integrate it into the commercial modeling tools MODELIO and MEDINI ANALYZE of the two involved project partners. In the first step towards analyzing and defining the viewpoint concept, a broad foundation of publications consisting of some 200 papers was built up. This was achieved by performing a search on the literature databases EBSCO Business Source Premier, Thomson Reuters Web of Knowledge and Google Scholar using search terms like ”viewpoint”, ”definition of viewpoint” or ”viewpoint-based modeling”. Afterwards, promising literature mainly focusing on viewpoint definitions have been picked up and screened to define our perspective on a general-applicable viewpoint definition. In order to assure an objective analysis, a framework consisting of eight categories has been constructed. To derive implications for modeling tools, literature mainly focusing on general and dedicated modeling features has been selected and analyzed by using two frameworks that have been created and which consist of nine and seven categories respectively. In this regard, we stress that some of these categories are taken from a number of viewpoint-related methods originated from the Software Engineering and Enterprise Architecture (EA) context.

To apply the proposed methodology, the remainder of this paper is organized as follows. Section 2 presents a discussion on the multi-domain purpose of viewpoints and outlines current limitations. In Section 3, viewpoints and adjacent concepts are defined and linked to an overarching metamodel. Furthermore, the paper's viewpoint concept is exemplified on the EA domain based on the TOGAF framework. Moreover, implications for tooling based on various features are illustrated. Finally, Section 4 summarizes the paper and provides an outlook to our future work on VBM.

2 Viewpoints—A Multi-Domain Concept and its Current Limitations

Without doubt, the concept of viewpoints is not novel. According to Lankhorst [Lan09], already in 1985, the *MultiView* approach proposed by Wood-Harper et al. [WHAA85] forms the concept's origin. *MultiView* aims at supporting the development process of (computerized) information systems by splitting its complex process into five different perspectives respective viewpoints: Human Activity System, Information Modelling, Socio-Technical System, Human-computer Interface, and Technical System. A decade later, the *MultiView* framework has been revised to *MultiView2* [AWHVW98]. Another early-published and frequently-cited work was conducted by Finkelstein et al. [FEKN92]. Even though it was published in the *International Journal of Software Engineering and Knowledge Engineering*, the focus of their work is not limited to software development, but can be applied to multiple kinds of artifacts which need to be constructed through a non-trivial engineering process. Hence, this corroborates the multi-domain spanning interest of the viewpoint concept, which will be further outlined in the remainder of this section.

With a stronger focus on software systems, Kruchten [Kru95] presents a model for describing the architecture of software-intensive systems, which it is often referred to as the *"4+1" View Model of Software Architecture*. This approach aims at the design and the implementation of the high-level structure of software. The architecture model is composed of multiple, concurrent perspectives called views. A view addresses a specific set of concerns looking on the system of the perspective of a particular stakeholder (group) [Kru95]. Also in the context of software development, viewpoints are utilized in the domain of requirements engineering. For example, Kotonya & Sommerville present a *Viewpoint-Oriented Requirements Definition* (VORD) approach in [KS96].

Another domain in which the viewpoint technique is frequently applied to is the Enterprise Architecture (EA) domain. In the course of this, the *Zachman Framework* [Zac00] (initial called as *Framework for Information Systems Architecture*), which is one of the earliest-published and most known frameworks, consists of a two dimensional classification matrix. The first dimension differentiates six different viewpoints: Planner, Owner, Designer, Builder, Subcontractor and Functioning System view. Orthogonal to this dimension, the Zachman framework differentiates between six different aspects: Data, Function, Network, People, Time and Motivation description. Another EA framework that is frequently used in practice is the *The Open Group Architecture Framework* (TOGAF) [The11]. TOGAF differentiates—without providing a concrete definition—between four viewpoints: Business, Data, Application and Technology Architecture.

While the previously mentioned concepts of viewpoints rather follow the idea of separating an artifact of interest in a vertical dimension meaning a union of all viewpoints would provide an overall vision on an artifact based on a specific level of detail, there is also a popular concept following the opposite direction [GBB12]. The *Model Driven Architecture* (MDA) [Obj03] follow the idea of introducing a strong separation of concerns regarding modeling a system at different abstraction levels. This starts from computational (CIM) and platform-independent (PIM) models and uses transformations to produce the actual code for the selected programming language and platform (PSM). In contrast to vertical-centric viewpoints, a viewpoint in MDA encompasses the whole underlying system, but on its specific level of detail regarding the distance to a concrete system implementation. Of course, even in MDA, a separation of concerns—preferable by using viewpoints—within one abstraction level seems necessary to reduce complexity.

Even though this section outlined the fact that viewpoints as a concept are very often used in multiple domains, the concept either lacks a definitional or scientific foundation (e.g. [The11]) or it strongly focuses on a specific domain. Furthermore, knowledge on requirements and implications for modeling tools aiming at realizing the viewpoint for practical usage concept is missing. One first approach towards this direction is the one recently published by Goldschmidt et al. [GBB12]. Even though their work is closely related to the one we present in this paper, it mainly focuses on a feature-based classification of view-based domain-specific modeling concepts, while in contrast we follow a more general approach; nevertheless, this recent paper shows that there is still a huge need in research particular regarding modeling tools that provide means for using the viewpoint technique.

3 The Concept of Viewpoints—Definition, Usage and Tool-Support

According to our observations, most of the definitions we found have something in common: they define the concept of ”viewpoint” as a guideline for constructing views. This can for example be observed in the *IEEE 1471:2000* standard definition [Sof07] for viewpoints, which was adopted by ISO/IEC as an *ISO/IEC 42010:2007* standard in 2007 [Hil11]. Accordingly, a viewpoint is ”a specification of the conventions for constructing and using a view”. This definition is most widely accepted in general system engineering, but also in software development. A related viewpoint definition is for example given by [ACWG94]. Consequently, the viewpoint can be seen as a pattern that defines a set of views. Another common feature of most of those definitions is that a viewpoint explicitly specifies one or more stakeholders, whose point of view it represents. Furthermore, some definitions explicitly note that a viewpoint should be as much self-contained as possible. What we are particularly interested in and will be investigating in this section is whether viewpoints are defined in terms of a metamodel, as well as whether this metamodel is separate or somehow related to other metamodels. Having relations or guidelines that drive viewpoint life-cycle and interaction as well as (de-)centralization of viewpoint underlying metamodels are further aspects in this section. Furthermore, another feature we are interested in is whether a viewpoint directly reflects the needs of a particular stakeholder.

3.1 Towards Defining the Viewpoint Concept

Due to the large amount of the related work and definitions found in the literature (more then a dozen of definitions and additionally methods for dealing with viewpoints providing their own definitions), we first clarify the terms and definitions which we take as a basis. These terms are stated in the lists below and visualized in Figure 1. We give a brief characterization of the basic terms where the first 5 below were derived and are in line with the definition given in OMG's Model Driven Architecture (MDA) [Obj03]:

Metamodel: A metamodel defines a frame and a set of rules for creating models for a specific application domain. Metamodels typically establish possible domain objects and relationships between them, as well as constraints that should be applied to them. Metamodels serve as a basis for models instantiating them, profiles referencing them and viewpoints that choose multiple modeling concepts from various metamodels to represent as a comprehensive big picture. Related modeling concepts usually belong to a certain metamodel.

Model Concept: A model concept is an element of a metamodel. The metamodel contains a type for each relevant modeling concept and defines the relation between these types. Model concepts are part of certain metamodel and are the basis for model elements in model instances as well as for the definition of viewpoints, which might be defined across multiple metamodels.

Model: A model is an instance of a metamodel. It contains a concrete set of model elements, which adhere to the rules defined in the corresponding metamodel. Models can apply certain profiles and thus represent model elements accordingly and serve as collection items for views.

Model Element: A model element is a concrete instance of a modeling concept, and thus it represents either a domain object or a relationship between two or more objects. These elements are a part of a certain model and are being exposed in certain views belonging to certain viewpoint instances.

Profile: A profile is an extension of a metamodel, which uses the metamodel as a reference for redefining existing modeling concepts and thus targeting specific domains. It serves a basis for viewpoints, refers to a certain metamodel and can be applied to various models for domain alignment.

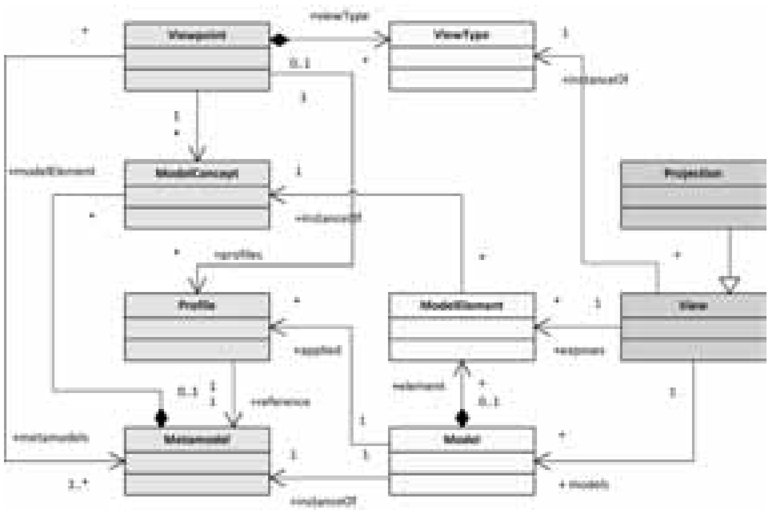


Figure 1: VIBAM's Position on Viewpoint Concept.

We give an additional list of important terms with brief characterizations where we give references to the work that was of major influence on our understanding of the meaning:

Viewpoint (cf. [The11], [Lan09], [CMR03]): The purpose of a viewpoint is to support a stakeholder in contributing to system design from a specific perspective. A viewpoint defines what concepts and relations can be used to define, view, or manipulate model instances within this viewpoint. It is therefore related to a (set of) metamodel(s), a (set of) profile(s) or a part of them. The viewpoint in this sense can restrict the original metamodel(s) but it can also correspond to a metamodel 1:1. The viewpoint is defined by the collection of view types that it offers to the stakeholders and which are instantiated by views.

View Type (cf. [GBB12]): A collection of view types is defined for each viewpoint. A view type serves a basis for view instantiation and offers a specific slice of system perspective to the stakeholders (i.e. human users).

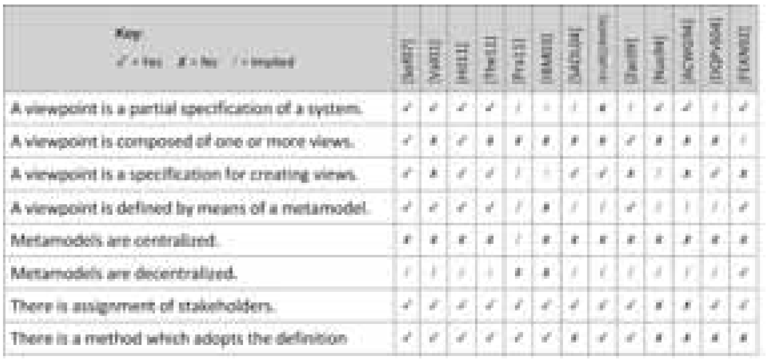
View (cf. [Lan09], [CMR03]): A view defines the presentation of model elements to a stakeholder and the way(s) how they can be modified (this is usually achieved by diagram types). It enables the user to interact with particular aspects of one or more models that adhere to the viewpoint's metamodel.

Projection (cf. [Pra11], [IBM03]): Function that maps a model instance M to another model instance M' where M' is a restriction of M in the sense that it contains only elements that are also contained in M and both M and M' are instances of the same metamodel. A projection is a special case of defining a view.

An intuitive reading of the metamodel depicted in Figure 1 is that viewpoints are either directly defined on metamodels or on top of profiles. Viewpoints offer a set of views which allow the stakeholder for whom the viewpoint was defined to access the model instances. Based on the general terms and the analysis of existing definitions, we derived a harmonized viewpoint definition that serves as a basis for the work in the VIBAM project.

Definition: A viewpoint is defined in relation to one or more metamodels. For each viewpoint a non-empty set of view types is defined. In a viewpoint instance any number of instance views for each of the view types can be dynamically created.

In the following we list features of viewpoint definitions which we consider important and which influenced our understanding on viewpoints during related work analysis. Moreover, Table 1 shows whether these features are considered within existing definitions (cf. [Sof07], [Val01], [Hil11], [The11], [Pra11], [IBM03], [SADL04], [Kru95], [RW05], [Zac09], [Nus94], [ACWG94], [DQPvS04], [FEKN92]).



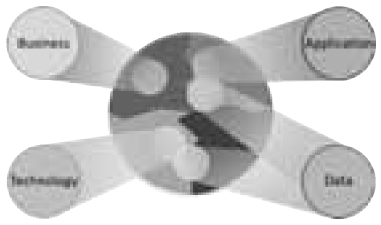
Key: ✓ = Yes ✗ = No / = Implied	[Sof07]	[Val01]	[Hil11]	[The11]	[Pra11]	[IBM03]	[SADL04]	[illegible]	[Zac09]	[Nus94]	[ACWG94]	[DQPvS04]	[FEKN92]
A viewpoint is a partial specification of a system.	✓	✓	✓	✓	/	/	/	✗	/	✓	✓	/	✓
A viewpoint is composed of one or more views.	✓	✗	✓	✗	✗	✗	✗	✗	✓	✗	✗	✗	/
A viewpoint is a specification for creating views.	✓	✗	✓	✓	/	/	✓	✓	✗	/	✗	✓	✗
A viewpoint is defined by means of a metamodel.	✓	✓	✓	✓	/	✗	/	/	✓	/	/	/	✓
Metamodels are centralized.	✗	✗	✗	✗	/	✗	✗	✗	✗	✗	✗	✗	✗
Metamodels are decentralized.	/	/	/	/	✗	✗	/	/	/	/	/	/	✓
There is assignment of stakeholders.	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✓	✓
There is a method which adopts the definition	✓	✓	✓	✓	✓	✓	✗	✓	✓	✗	✗	✗	✗

Table 1: Viewpoint Definition Features.

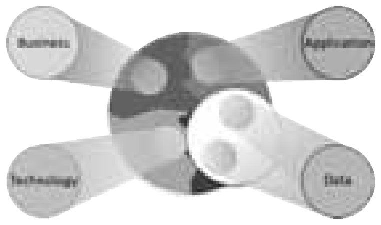
A viewpoint is a partial specification of a system: A viewpoint contains certain functional description and information types which are implemented and used in the developed system at run-time. As we can see from Table 1, almost all of the research shares this opinion over the viewpoint definition. *A viewpoint is composed of one or more views:* A viewpoint is defined by a language which we can refer to as a metamodel, and it explicitly addresses specific stakeholders. Only a few authors (IEEE [Sof07] and ISO [Hil11] standards, Zachman [Zac09] and Finkelstein et al. [FEKN92]) use this notion of the viewpoint in their definitions, whereas this leads us to a more comprehensive and consistent system description. *A viewpoint is a specification for creating views:* A viewpoint is a pattern or template from which to develop individual views. Thus a view is a concrete instance of a viewpoint. Most of the authors see the viewpoint as this, whereas specification of views as view typing is how the authors of this paper understand this term. *A viewpoint is defined by means of a metamodel:* A viewpoint is a type of metamodel for view creation. As in the previous point, almost everybody agrees on metamodels being a basis for viewpoints defining their target domain usage. *Metamodels are centralized:* This means there is no clear separation between viewpoints and views. Each viewpoint governs which kind of model element can be represented, the consistency rules and completeness rules that needs to be applied, and the different view that can be provided. Only PRAXEME [Pra11] method partly defends this point of view. *Metamodels are decentralized:* This means a viewpoint is a loosely coupled, locally managed, coarse-grained object which encapsulates partial knowledge about the system and domain, specified in a particular, suitable representation scheme. Apart from PRAXEME [Pra11] and RUP [IBM03] other methods are not supporting this point of view. *There is assignment of stakeholders:* Each view targets a specific group of stakeholders, thus separating modeling concerns and assuring consistency. Each stakeholder is then responsible for designing his model part with the aid of constructs provided by the assigned view. This is agreed upon by everybody except for Ainsworth et al. [ACWG94] and Nuseibeh et al. [Nus94]. *There is a method which adopts the definition:* Existence of a method in research or industry, which uses the current viewpoint definition, is under question here. If a viewpoint definition is not used in any methods, this makes the definition a pure academic matter.

3.2 Applying Viewpoints on TOGAF

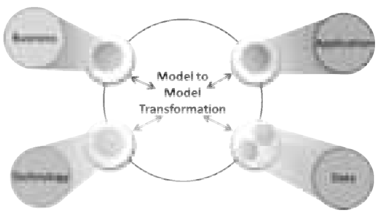
In order to clarify our theoretical explanations, this section will exemplify our understanding of the viewpoint concept on the Enterprise Architecture domain using the TOGAF framework as a basis. As already mentioned in Section 2, TOGAF differentiates between four different viewpoints. Figure 2a shows the different stakeholders regarding the TOGAF method for system design. When we want to deal with viewpoint concepts in a technical sense, we can start off with looking at an application domain for which we capture the relevant concepts in one or more metamodels. We refer to the set of these basic metamodels with $\mathcal{M}_b$ (displayed in blue in Figure 2a). To define a viewpoint we select the set of concepts from $\mathcal{M}_b$ that we consider relevant for the viewpoint (see Figure 2b). The selection criteria is whether a specific concept is relevant for the stakeholder which the given viewpoint should then support. The selected concepts form the concepts for the viewpoint metamodel M_v. The relations for all concepts in M_v have to reflect the relations between the corresponding concepts in $\mathcal{M}_b$. We do assume that the relations between concepts in $\mathcal{M}_b$ do not contradict each other.



(a)



(b)



(c)

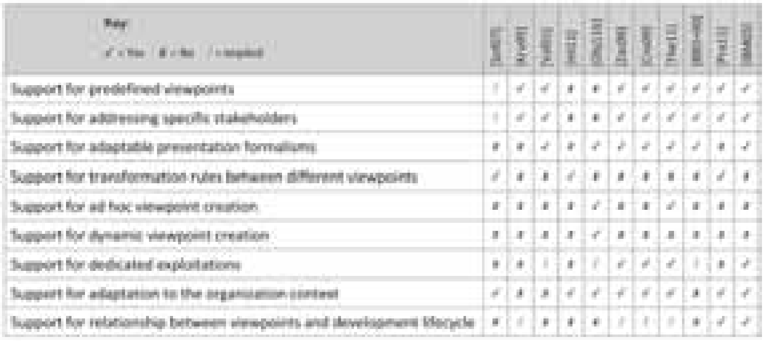
Figure 2: TOGAF Perspective to System Design and Information Exchange between Viewpoints.

For a viewpoint metamodel M_v we now can define concrete views that will allow the stakeholder to actually access the model instances that adhere to the viewpoint metamodel M_v. Model to model transformations are used to pass information or model fragments between different viewpoints (see Figure 2c). OCL constraints included in the metamodel representations are used to validate model instances. Even within a viewpoint definition given by a viewpoint metamodel M_v projections can be used to distinguish between different stakeholders and with this support them with different views to the model instances. Projections are defined on the model instance level and can be even defined dynamically by attaching annotations to the model instances. There is no real need to make a copy of a model instance to create a new projection. Rather the defined views are adapted or restricted in a manner that serves the respective stakeholder best. If more than one human user is manipulating the same model instance possibly using different perspective (i.e. by using different projections), consistency of the model instance is of major concern.

3.3 Implications for Tooling

The features of viewpoints given in this section were derived with respect to VIBAM's viewpoint definition to allow for a comparison of viewpoint definitions of relevant methods from literature. In the following we present the comparison of the methods which we consider most important: IEEE 1471-2000 [Sof07], Kruchten [Kru95], RM-ODP [Val01], ISO/IEC 42010 [Hil11], SysML [Obj11], Zachman [Zac09], MODAF [Cro09], TOGAF [The11], Boiten [BBD$^+$00], PRAXEME [Pra11], RUP [IBM03]. The selected viewpoint methods have been examined from two angles: general and dedicated features. The difference between them lies in the scope of the analysis of the viewpoint methods under surveillance. The general features refer rather to the external influence on the viewpoints like support for predefined viewpoints and transformation rules between them. Dedicated features refer to the internal features of the viewpoints like viewpoint definition, consistency rules inside and between viewpoints. A closer look at the analysis tables below gives insight into the tooling features we derived and recommend for further implementation.

General Viewpoint Features of the Methods



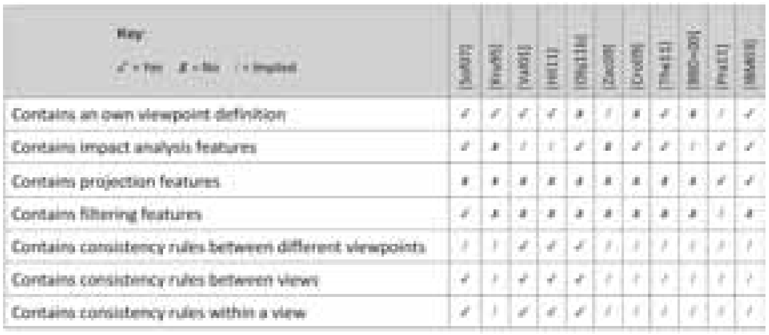
Key: ✓ = Yes ✗ = No ! = Implied

	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Support for predefined viewpoints	!	✓	✓	✗	✗	✓	✓	✓	✓	✓	✓
Support for addressing specific stakeholders	!	✓	✓	✗	✗	✓	✓	✓	✓	✓	✓
Support for adaptable presentation formalisms	✗	✗	✓	✗	✓	✓	✓	✓	✓	✗	✓
Support for transformation rules between different viewpoints	✓	✗	✗	✓	✗	✗	✗	✗	✗	✓	✗
Support for ad hoc viewpoint creation	✗	✗	✗	✗	✓	✗	✗	✓	✗	✗	✗
Support for dynamic viewpoint creation	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
Support for dedicated exploitations	✗	✗	!	✗	!	✓	✓	✓	!	✗	✓
Support for adaptation to the organization context	✓	✗	✗	✓	✓	✓	✓	✓	✗	✓	✓
Support for relationship between viewpoints and development lifecycle	✗	!	✗	✗	✗	!	!	!	✗	✓	✓

Table 2: Viewpoint General Features.

Support for Predefined Viewpoints: A predefined viewpoint is defined prior to the application of a method. Usually, there is a fixed set of predefined viewpoints unrelated to any domain, i.e. they are for example defined by the method itself (cf. RM-ODP). As a consequence, this feature can imply a limitation on versatility of the method's application. One example in this regard is TOGAF which is a dedicate enterprise architecture. Apart from ISO/IEC and SysML, all of the methods are defining viewpoints in advance. *Support for Addressing Specific Stakeholders:* This feature defines whether the method targets specific stakeholders and hence proposes specific concepts for these stakeholders. This implicates a number of predefined user groups, which in turn means targeting specific domains in advance resulting in limitation of versatility and more specific stakeholder targeting. As a result, all considered methods except for ISO/IEC and SysML are defining stakeholders a priori. *Support for Adaptable Presentation Formalisms:* Adaptable presentation formalisms provide the ability to adjust the presentation of model elements to the needs of certain users or to conform to certain viewpoints. This can be realized by a profile. Apart from IEEE, Kruchten, ISO/IEC and PRAXEME, all methods allow a flexible definition of views. *Support for Transformation Rules between different Viewpoints:* This feature expresses the existence of constructive rules that allow deriving model elements for a particular viewpoint out of model elements from another viewpoint. These rules can be understood as model transformation between different viewpoints. Although almost all of the viewpoint definitions support decentralized viewpoints, not every method provides transformation rules—except for IEEE, ISO/IEC and PRAXEME. *Support for Ad Hoc Viewpoint Creation:* Some methods recommend to define viewpoints during a project's preparation phase to address unforeseen stakeholder groups. This feature is called ”ad hoc” in contrast to ”predefined” viewpoints. However, only SysML and TOGAF see the need for making room for such developments. *Support for Dynamic Viewpoint Creation:* Creating a viewpoint completely dynamically in a sense of creating a viewpoint on the fly after a project has already begun. SysML is the only method that provides means for giving stakeholders this capability. *Support for Dedicated Exploitation:* A certain need for a usage of a viewpoint unintended during the design time may occur at run-time (e.g. a viewpoint with deactivated dependencies to other viewpoints for protecting sensible information). Most of the methods support this at least partially. *Support for Adaptation to the Organization Context:* This feature defines whether the existing viewpoint of a method can be adapted to a specific organizational context in order to suit the intended usage of the system. The three methods—Kruchten, RM-ODP and Boiten—do not see the need for adapting the viewpoints to specific organizations. *Support for Relationship between Viewpoints and Development Lifecycle:* This features outlines whether the method provides certain guidance or recommendations to relate viewpoints with the development lifecycle of systems. This is not supported by all of the methods—only two of them, namely PRAXEME and RUP, are looking into realizing this feature to the full extent.

Dedicated Viewpoint Features of the Methods



Key: ✓ = Yes ✗ = No ! = Implied	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
Contains an own viewpoint definition	✓	✓	✓	✓	✗	!	✗	✓	✗	!	✓
Contains impact analysis features	✓	✗	!	!	✓	✗	✓	✓	!	✓	✓
Contains projection features	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓
Contains filtering features	✓	✗	✗	✗	✗	✗	✗	✗	✗	!	✗
Contains consistency rules between different viewpoints	!	!	✓	✓	✓	!	!	!	!	!	!
Contains consistency rules between views	✓	!	✓	✓	✓	!	!	!	!	!	!
Contains consistency rules within a view	✓	!	✓	✓	✓	!	!	!	!	!	!

Table 3: Viewpoint Dedicated Features.

Contains an own viewpoint definition: This feature signifies whether the method provides its own viewpoint definition. SysML, MODAF and Boiten do not provide own definitions. *Contains impact analysis features:* The question here is whether a traceability methodology is available and, if yes, whether it allows impact analysis of the model changes between viewpoints. All the methods except for Kruchten and Zachman do explicitly state this. *Contains projection features:* This feature is derived from the projection definition in Section 3.1. Hence, the idea is to let different viewpoints edit the same model, whereas certain constructs are represented in different ways in each of the viewpoints. None of the methods apart from PRAXEME and RUP adhere to this feature. *Contains filtering features:* In their filtering capacities, viewpoints will filter out those model elements that are not allowed to appear under a certain view, and thus only elements eligible for a viewpoint will be provided for view modeling. The two methods IEEE and PRAXEME are at least partially support this capability. *Contains consistency rules between different viewpoints:* If a system is modeled using different viewpoints, the model elements which are defined in these viewpoints are usually not completely independent form each other. There might be certain rules that need to be obeyed to ensure the overall consistency of the underlying model. All of the methods support the consistency rules feature. *Contains consistency rules between views:* In the same manner as it was for consistency between different viewpoints, certain rules may need to be obeyed in order to ensure the consistency in a certain viewpoint between its views. All of the methods ask for consistency between the views. *Contains consistency rules within a view:* As in the two features before, there might be a threat for inconsistencies inside a certain view due to editing from the different view instances. As a consequence, certain instance level consistency rules may have to be obeyed. All methods require this feature.

4 Conclusion and Future Work

In this paper we reported on results of the VIBAM project in which viewpoint concepts are investigated. We presented an overview of the current state-of-the-art for viewpoint definitions, concepts and methods. Derived from the definitions we found in literature we present definitions for the list of concepts on the basis of which we define the position that we take in the VIBAM project regarding viewpoint concepts. After we discussed VIBAM's position on viewpoints we presented a list of features that we consider important for tool support of the viewpoint definition presented in this paper.

The next step in our work is to integrated the defined concepts in the commercial modeling tools MODELIO and MEDINI ANALYZE. With respect to the basic technologies the implementation of dynamic viewpoint creations is rather difficult to achieve. Even the definition of views for example on the basis of EMF/GMF is cumbersome if one is not satisfied with the default behavior that is offered for this technology stack. We will investigated what changes would be needed to make the use of the basic technologies more flexible.

Acknowledgements: This work has been funded by the German Federal Ministry of Education and Research (FKZ 01QE1106B/C) and by the French OSEO through ICT Project ViBaM (Viewpoint-Based Modeling) which is running in the context of the European Eurostars Program (E!5529). The authors wish to acknowledge EUREKA and the Commission for their support. We also thank the reviewers for their valuable comments.

References

[ACWG94] M. Ainsworth, A.H. Cruickshank, P.J.L. Wallis, and L.J. Groves. Viewpoint specification and Z. In *Information and Software Technology*, 36(1):43–51, 1994.

[AWHVW98] D.E. Avison, A.T. Wood-Harper, R.T. Vidgen, and J.R.G. Wood. A further exploration into information systems development: the evolution of Multiview2. In *Infor-*

mation Technology & People, 11(2):124–139, 1998.

[BBD+00] E. Boiten, H. Bowman, J. Derrick, P. Linington, and M. Steen. Viewpoint consistency in ODP. In *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 34(3):503–537, 2000.

[BKS10] S. Buckl, S. Krell, and C.M. Schweda. A Formal Approach to Architectural Descriptions—Refining the ISO Standard 42010. In A. Albani, J.L.G. Dietz, W. van der Aalst, J. Mylopoulos, M. Rosemann, M.J. Shaw, and C. Szyperski, editors, *Advances in Enterprise Engineering IV*, Vol. 49 of *Lecture Notes in Business Information Processing*, pp. 77–91. Springer, Berlin Heidelberg, Germany, 2010.

[BW06] P. Balabko, and A. Wegmann. Systemic classification of concern-based design methods in the context of enterprise architecture. In *Information Systems Frontiers*, 8(2):115–131, 2006.

[CMR03] J. Champeau, F. Mekerke, and E. Rochefort. Towards a Clear Definition of Patterns, Aspects and Views in MDA. In *Proc. of the First International Workshop on Engineering Methods to Support Information Systems Evolution*, Geneva, Switzerland, 2003.

[Cro09] Crown. A summary of MODAF views by their use and data types, 2009.

[DQPvS04] R.M. Dijkman, D.A.C. Quartel, L.F. Pires, and M.J. van Sinderen. A rigorous approach to relate enterprise and computational viewpoints. In *Proc. of the 8th IEEE Enterprise Distributed Object Computing (EDOC) Conference*, Monterey, California, USA, 2004.

[DQvS08] R.M. Dijkman, D.A.C. Quartel, and M.J. van Sinderen. Consistency in multi-viewpoint design of enterprise information systems. In *Information and Software Technology*, 50(7-8):737–752, 2008.

[EYA+05] S. Easterbrook, E. Yu, J. Ar, Y. Fan, J. Horkoff, M. Leica, and R.A. Qadir. Do viewpoints lead to better conceptual models? An exploratory case study. In *RE*, pp. 199–208, 2005.

[FEKN92] A. Finkelstein, S. Easterbrook, J. Kramer, and B. Nuseibeh. Requirements Engineering Through Viewpoints. In *DRA Colloquium on Analysis of Requirements for Software Intensive Systems*, pp. 18–26, Defence Research Agency, 1993.

[FKN+92] A. Finkelstein, J. Kramer, B. Nuseibeh, L. Finkelstein, and M. Goedicke. Viewpoints: A Framework for Integrating Multiple Perspectives in System Development. In *International Journal of Software Engineering and Knowledge Engineering*, 2(1):31–57, 1992.

[GBB12] T. Goldschmidt, S. Becker, and E. Burger. Towards a Tool-Oriented Taxonomy of View-Based Modelling. In E.J. Sinz, and A. Schürr, editors, *Modellierung*, Vol. 201 of *GI-LNI*, pp. 59–74, 2012.

[Hil11] R. Hilliard. Welcome to the ISO/IEC 42010 Website. `http://www.iso-architecture.org/ieee-1471/`, Accessed July 2012.

[IBM03] IBM. Rational Unified Process: A Best Practices Approach, 2003.

[Kru95] P. Kruchten. Architectural Blueprints - The 4+1 View Model of Software Architecture. In *IEEE Software*, 12(6):42–50, 1995.

[KS96] G. Kotonya, and I. Sommerville. Requirements Engineering With Viewpoints. In *Software Engineering Journal*, 11(1):5–18, 1996.

[Lan09] M. Lankhorst. *Enterprise Architecture at Work: Modelling, Communication and Analysis*. Springer, Berlin Heidelberg, Germany, 2009.

[MP00] R. Motschnig-Pitrik. The viewpoint abstraction in object-oriented modeling and the UML. In *Proc. of the 19th international conference on Conceptual modeling*, ER'00, Salt Lake City, Utah, USA, 2000.

[Nus94] B.A. Nuseibeh. *A Multi-Perspective Framework for Method Integration*. Dissertation, Imperial College of Science, Technology and Medicine, University of London, Department of Computing, 1994.

[Obj03] Object Management Group. MDA Guide Version 1.0.1. `http://www.omg.org/cgi-bin/doc?omg/03-06-01`, Accessed July 2012.

[Obj11] Object Management Group. The Official OMG SysML website. `http://www.omgsysml.org`, Accessed July 2012.

[Pra11] Praxeme Institute. PRAXEME - Opus, the Product. `http://www.praxeme.org/index.php?n=Opus.Opus`, Accessed July 2012.

[RW05] N. Rozanski, and E. Woods. *Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives*. Addison-Wesley Longman, Amsterdam, 2005.

[SADL04] M.W.A. Steen, D.H. Akehurst, H.W.L. Doest, and M.M. Lankhorst. Supporting Viewpoint-Oriented Enterprise Architecture. In *Proc. of the 8th IEEE International Enterprise Distributed Object Computing Conference (EDOC'04)*, Monterey, California, USA, 2004.

[Sof07] Software Engineering Standards Committee of the IEEE Computer Society. IEEE Recommended Practice for Architectural Description of Software-Intensive Systems. Software Engineering Standards Committee of the IEEE Computer Society, 2007.

[Sta73] H. Stachowiak. *Allgemeine Modelltheorie*. Springer, Berlin Heidelberg, Germany, 1973.

[The11] The Open Group. Welcome to TOGAF Version 9 - an Open Group Standard. `http://pubs.opengroup.org/architecture/togaf9-doc/arch/index.html`, Accessed July 2012.

[Val01] A. Vallecillo. RM-ODP: The ISO Reference Model for Open Distributed Processing. *DINTEL Edition on Software Engineering*, 3:66–69, 2001.

[WHAA85] A.T. Wood-Harper, L. Antill, and D.E. Avison. *Information systems definition: the Multiview approach*. Blackwell Scientific Publications, Oxford, UK, 1985.

[Zac00] J.A. Zachman. Concepts of the Framework for Enterprise Architecture. `http://www.ies.aust.com/papers/zachman3.htm`, Accessed July 2012.

[Zac09] J.A. Zachman. The Zachman Framework: The Official Concise Definition. `http://old.zachmaninternational.com/concise%20definition.pdf`, Accessed July 2012.