Towards Utility-based Selection of Architecture-Modelling Concepts

H.A. Proper[1], A.A. Verrijn-Stuart[2] and S.J.B.A. Hoppenbrouwers[1]

[1] University of Nijmegen*, Sub-faculty of Informatics, IRIS Group, Toernooiveld 1, 6525 ED Nijmegen, The Netherlands, EU e.proper@acm.org, stijnh@cs.kun.nl

[2] Emeritus of the University of Leiden, Faculty of Mathematics and Science, LIACS, Scheltemakade 15 (home), 2012 TD Haarlem, The Netherlands, EU Alexander.VerrynStuart@wxs.nl

Abstract. In this paper we are concerned with the principles underlying the utility of modelling concepts, in particular in the context of architecture-modelling. Firstly, some basic concepts are discussed, in particular the relation between information, language, and modelling. Our primary area of application is the modelling of enterprise architectures and information system architectures, where the selection of concepts used to model different aspects very much depends on the specific concerns that need to be addressed. The approach is illustrated by a brief review of the relevant aspects of two existing frameworks for modelling of (software intensive) information systems and their architectures.

1 Introduction

The importance of information systems to modern day society needs no argueing. Information systems may range from small-scale systems geared towards a few users, via systems supporting the tasks of a business unit, to enterprise-wide systems and even value-chain wide systems. The ubiquity of information systems, combined with the high levels of integration with our daily lifes, be it at work or during leisuretime, puts high demands on the development processes of these systems.

Information systems, as their name suggests, primarily handle 'information'. Based on the definition provided in [FHL+98], we define *information* to be: the *knowledge* increment brought about when an *actor* receives a message. A direct consequence of this definition is that we regard messages that result from an information system as representations of knowledge. In line with [FHL+98, BMS98] we consider an *information system* to be a system for collecting, processing, storing, retrieving and distributing information within an organisation and between the organisation and its environment. As such, an information system can be regarded as a subsystem of the organisation (focussing on the informational aspects of an organisation), and may consist of both *human* and *computerised* actors.

In the last decennium, several approaches to the development of larger (anything beyond small scale systems geared towards a few specific users) information systems have emerged that depend highly on the use of so-called '*architectures*' [Zac87, Boa99, BMS98]. Some of these approaches use the term 'information architecture', or 'architecture of information systems', while yet others refer to the same concept as 'enterprise (IT) architecture'.

In [IEE00], the concept of architecture is defined as: "*The fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution.*". Architectures are usually expressed in terms of architectural descriptions, essentially design descriptions pertaining to the architecture of a system. In general, the rationale behind the use of architecture in the context of information systems is that it provides a number of important benefits [BCK98, IEE00], such as:

* This paper results from the ArchiMate project (http://archimate.telin.nl), a research consortium that aims to provide concepts and techniques to support enterprise architects in the visualisation, communication and analysis of integrated architectures.

– It is a vehicle for communication and negotiation among stakeholders. A software architecture, often depicted graphically, can be communicated with different stakeholders involved in the development, production, fielding, operation, and maintenance of a system.
– It captures essential design decisions, both functional aspects as well as quality aspects. In an architecture, the global structure of the system has been decided upon, while responsibilities (such as functionality) have been assigned to the (overall) components of the system.

In the conceptual framework for architecture, as defined in [IEE00], an architectural description can be organised into one or more constituents called architectural views. Each view addresses one or more of the *concerns* (interests) of the stakeholders of a system. The term 'view' is used to refer to the expression of a system's architecture with respect to a particular viewpoint. A viewpoint establishes the conventions by which a view is created, depicted and analyzed. In other words, a viewpoint determines the languages to be used to describe the view, and any associated modelling methods or analysis techniques to be applied to these representations of the view. These languages and techniques are used to yield results relevant to the concerns addressed by the viewpoint.

In the context of architectural descriptions, a plethora of frameworks of viewpoints is in existance, leaving designers and architects with the burden of selecting the viewpoints to be used in a specific situation. Some of these frameworks of viewpoints are: The Zachman framework [Zac87], Kruchten's 4+1 framework [Kru95], RM-ODP [ISO98b], ArchiMate [JBA+03] and TOGAF [The04]. The aim of this paper is not to provide 'yet another framework of viewpoints', but rather to lay a foundation to be able to *reason about such frameworks at a meta-level*. In other words, a fundamental approach is proposed allowing designers and architects to consider the relevance of specific viewpoints regarding their practical design/development tasks. Instead of making superficial comparisons between the specific abilities of various techniques, we aim at finding deeper motivations for the differences between them.

This paper is a product of an ongoing research activity which aims to gain a more fundamental understanding of the act of modelling, in the context of system development, and the languages that are used in the process. The view presented is developed via three complementary angles:

Modelling: This angle aims to provide a fundamental grounding of (architectural) modelling and representation. We will focus primarily on the foundations of modelling, representation of models and the role of languages.

Utility: The potential utility that specific modelling concepts may have when used to express architectural descriptions, from the perspective of a given design/development task.
This angle, which builds on the previous one, aims to provide designers and architects with the insights to reason about the relevance of modelling concepts to a specific task in the design/development process.

Communication: The (interpersonal) communication about architectural descriptions as it occurs during modelling and design.
This angle, further adding to the insights of the previous two, focuses on the role of architectural descriptions as a means of communication between a system's stakeholders, i.e. language in action.

Our study was directed primarily at modelling of architectures. The results, however, may equally well be applied to other areas of model-driven design, such as software design, business-process (re)design, city planning, architecting of buildings, etc.

We have structured the remainder of this paper by providing a discussion of the above three angles in separate sections (sections 2 to 4). To make our results more concrete, section 5 briefly discusses two example frameworks of viewpoints (Kruchten's 4+1 framework [Kru95] and RM-ODP [ISO98b]), from the perspective of our meta-framework. This is followed by a brief discussion on directions of further research and elaboration in section 7.

2 Modelling

The aim of this section is to investigate the process of modelling, as it occurs in e.g. architecture-modelling, more closely. In defining more precisely what we mean by modelling a domain, we first

need to introduce a framework describing the essential processes that take place when a viewer (such as a stakeholder) observes a domain (such as a system being developed).

Let us first consider what happens if some *viewer* observes 'the universe'. It is our assumption, based on the work of C.S. Peirce [Pei69a, Pei69b, Pei69c, Pei69d], that viewers perceive a universe and then produce a *conception* of that part they deem relevant. The conceptions harboured by a viewer are impossible to communicate and discuss with other viewers unless they are articulated somehow (the need for this ability in the context of system development is evident). In other words, a *conception* needs to be *represented*. Peirce argues that both the perception and conception of a viewer are strongly influenced by their interest in the observed universe. This leads to the following (necessarily cyclic, yet irreflexive) set of definitions:

Universe – the 'world' around the viewer.
Viewer – an actor perceiving and conceiving the universe, using their senses.
Conception – that which results, in the mind of a viewer, when they observe the universe, using their senses, and interpret what they perceive.
Representation – the result of a viewer denoting a conception, using some language and medium to express themselves.

The underlying relationships between viewers, universe, conceptions and representations can be expressed in terms of the so-called FRISCO tetrahedron [FHL+98], as depicted in figure 1.

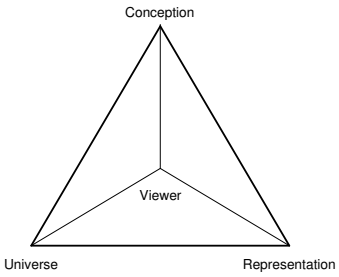

Fig. 1. The FRISCO tetahedron.

As mentioned above, in conceiving a part of the universe, viewers will be influenced by their particular interest in the observed universe. In the context of system development, this corresponds to what tends to be referred to as a *concern*. For example:

- The current situation with regard to the computerised support of a business process.
- The requirements of a specific stakeholder with regard to the desired situation.
- The potential impact of a new system on the work of the system administrators that are to maintain the new system.

Concerns are not the only factors that influence a viewers conception of a domain. Another important factor are the pre-conceptions a viewer may harbour as they are brought forward by their social, cultural, educational and professional background. More specifically, in the context of architecture modelling, viewers will approach a domain with the aim of expressing the domain in terms of some set of meta-concepts, such as classes, activities, constraints, etc. The set of meta-concepts a viewer is used to using (or trained to use) when modelling a domain, will strongly influence the conception of the viewer. This can be likened to the typical situation of having a 'hammer' and considering all pointy objects to be 'nails'. We therefore presume that when viewers model a domain, they do so from a certain perspective; their *weltanschauung* (German for "view of the world") [WAA85]. The weltanschauung can essentially be equated to the notion of a *viewpoint* as discussed in section 1. This perspective on the notion of viewpoints is compatible to the approach taken in the Reference Model of Open Distributed Processing [ISO98b]:

"In order to represent an ODP system from a particular viewpoint it is necessary to define a structured set of concepts [the meta-concepts] in terms of which that representation (or specification) can be expressed. This set of concepts provides a language for writing specifications of systems from that viewpoint, and such a specification constitutes a model of a system in terms of the concepts."

In general, people tend to think of the universe (the 'world around us') as consisting of related *elements*. In our view, however, presuming that the universe consists of a set of elements already constitutes a subjective choice, which essentially depends on the viewer observing the universe. The choice being made is that 'elements' (or 'things') and 'relations' are the most basic concept for modelling the universe; the most basic weltanschauung. In the remainder of this paper, we will indeed make this assumption, and presume that a viewer's *conception* of the universe consists of elements. The identification of elements in the universe remains relative to viewers and their *own* conception.

Viewers may decide to zoom in on a particular part of the universe they observe, or to state it more precisely, they may zoom in on a particular part of *their* conception of the universe. This allows us to define the notion of a domain as:

Domain – any subset of a conception (being a set of elements) of the universe, that is conceived of as being some 'part' or 'aspect' of the universe.

In the context of (information) system development, we have a particular interest in unambiguous abstractions from domains. This is what we refer to as a *model*:

Model – a purposely abstracted and unambiguous conception of a domain.

Note that both the domain and its model are *conceptions* harboured by the same viewer. We are now also in a position to define more precisely what we mean by modelling:

Modelling – The act of purposely abstracting a model from (what is conceived to be) a part of the universe.

For practical reasons, we will understand the act of *modelling* to also include the activities involved in the *representation* of the model by means of some language and medium.

We presume a viewer not only to be able to represent (parts of) their conceptions of the universe, but also to be able to represent (parts of) the viewpoints they use in producing their conception of the universe. This does require viewers to be able to perform some kind of self-reflection. When modelling some domain in terms of, say, UML class diagrams [BRJ99], the viewer/modeller is presumed to be able to express the fact that they are using classes, aggregations, associations, etc, to view the domain being modelled. In doing so, viewers essentially need to construct a conception of their viewpoint on the world; i.e. a meta-model. This meta-model comprises the meta-concepts and modelling approach used by the viewer when modelling a domain; it is a model of the viewers viewpoint. Such a meta-model can in essence be regarded as a 'high level ontology' [KZR04] as well.

In figure 2 we have depicted a situation where a viewer is confronted with a number of domains $(W_1, \ldots, W_n)$. Each of these domains may be modelled from the perspective of the viewer's concern $\underline{C}$ and meta-model $\underline{M}$, leading to even so many domain-models $(\underline{D}_1, \ldots, \underline{D}_n)$. The concern, the meta-model and the domain models can be represented using some language and medium, leading to representations $C, M, D_1, \ldots, D_n$.

A viewer may also consider a specific domain W from the perspective of some concern C, using two different meta-models M_1 and M_2. This situation is illustrated in figure 3, where a viewer models a domain D from the perspective of meta-models M_1 and M_2, leading to domain-models D_1 and D_2 respectively. For example, when viewing a domain from the perspective of UML class diagrams, this is bound to lead to a different domain-model than when the same domain is viewed from the perspective of UML sequence diagrams.

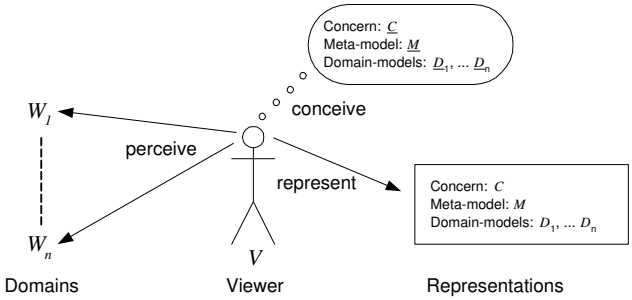

Fig. 2. A viewer viewing domains from a particular concern and meta-model.

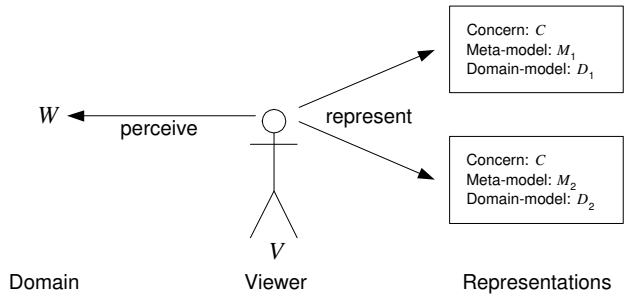

Fig. 3. A viewer viewing a domain from the perspective of two different meta-models.

If a viewer observes a domain D from the same meta-model M, but from the perspective of different concerns C_1 and C_2, it is also quite likely that the viewer will produce different domain-models, each catering to the specificities of the two specific concerns. Consider for example, a concern focussing on the functionality offered by a system to its users, versus a concern focussing on the impact of the system on the efficiency of business processes.

Given two different concerns, it is also likely that questions underlying these concerns cannot be met by using a one-size-fits-all meta-model. For example, the operators who will be required to maintain a planned information system, will regard this system in terms of costs of keeping the system up and running, costs and efforts involved in implementing the system, etc. Future users of the same planned system, however, will be more interested in the impact/support the system is likely to have on their work related tasks. This implies that when modelling a system (being designed/developed), different meta-models need to be used to address different concerns.

The focus of the next section will be to gain insight into the potential utility of a meta-concepts to a given modelling goal and viewer's concern.

3 Utility of modelling concepts

A fundamental problem to be addressed is the 'utility' of modelling concepts relative to some concern and modelling goal. Utility must be understood in the sense of its classical economic context, such as "what benefit do I derive from using it?" or "what use is it to me?". Our primary concern is in the area of information systems, of which both the modelling and model usage aspects must be considered. The former pertain to the expressiveness of modelling languages, the latter to the effectiveness of the ultimate system.

3.1 Representational economics

Intuitively, information is linked to knowledge (knowing how to do something, how to do it better, how to do it more timely, or how to avoid something). In other words, information is an important

ingredient of decision making. As mentioned above, a sensible definition is to equate 'information' with an increment in knowledge [FHL+98]. Value of information (or its 'utility') should then be associated with the advantage of better decision making and more effective 'goal-pursuit' [FHL+98, Section 2.5 and Chapter 3].

It is evident that the 'economics of information' are complex and cannot be derived from – if at all associated with – some identifiable and coherent market. While everyone will agree that something referred to as 'information' must have value, the contexts will vary so much that any pretense to a straightforward theory should be rejected. One of the reasons is that what we actually deal with are *representations* of domains in the real or imaginary world (i.e. 'models'), and of things, relationships and actions pertaining to those domains (i.e. 'data'). The theme of this paper is the utility of *how* and *what* one models. Regarding the link with the economics of information we will merely assert three things:

- a computerised system capable of providing *information* is of *value* to its owner/user;
- the *cost* of building and maintaining such a system may exceed its *value addition*;
- any means of *reducing cost* and/or *increasing value* in this context are desirable.

Since the systems we wish to construct are dynamic representations of a domain of interest, a key question is how best to describe such domains. What we look for is a meta-model which is simultaneously simple and rich. For instance, it must be capable of describing anything requiring modelling in our domain at all. On the other hand, it must be so restricted that the full range of concepts may be grasped by any individual modeller while simultaneously usable in exchanges between such a person and the many other interested parties concerned.

The solution to this problem lies in a judicious selection of 'concepts'. These should be both operationally effective and domain-encompassing. Effectiveness means having a high 'utility', that is to say, they must be generally accepted and permit 'economical' application. Encompassing one particular type of domain, but not necessarily all conceiveable ones, means that a 'goal-bounded' approach should be adopted whereby the modelling needs are restricted according to the modelling goal at hand. The modelling goal may differ from situation to situation. The problem, therefore, evolves to that of agreeing on and maintaining a large collection of concepts suitable to cover a range of domains and a systematic means of bounding it to fit any selected occasion.

The best way of achieving this is to resort to a set of high-level 'meta-concepts', which are capable of being specialised to fit contingencies. How general and large should that set be, what should be in it and, most importantly, how do we effectively and economically restrict it to cope with specific domains?

3.2 Meta-concepts and concept restrictions

One way of dealing with complexity is to abstract its features and describe a given situation in general terms. For instance, if an extended set of concepts is required, then grouping them in broad categories aids in better understanding them. Similarly, when in spite of all simplification efforts a complex language remains necessary, discussing and describing it in a 'meta-language' is often fruitful.

Thus, the problem of arriving at a 'goal-boundable' approach to modelling is to conceive an extremely simple *set of meta-concepts*, each of which is capable of being specialised so as to be used in a particular case. Now, information systems – which serve organizations – have two general aspects, both of which need to be captured in any model:

- Informational aspect – i.e. 'what to describe', but also, 'what to leave out'.
- System aspect – i.e. a cohesion-oriented 'description format'.

While the first gives rise to the *elements* of the ultimate system, the latter provides the *formal basis* for putting them together. An 'information system' is a special kind of 'system' and a 'system', in turn, is a specialisation of a 'model' [FHL+98]. Therefore, the concepts appropriate for underpinning an information system description must, after all, derive from the most general

concepts for modelling. These also constitute our desired set of 'meta-concepts'. The information and system aspects characterise the special domain 'information system' and, hence, may be covered under the umbrella 'domain-concepts' (covering all elementary and structural features of information systems, as such).

To summarize, our aim must be to devise a complete – and ideally minimal – set of *meta concepts* that are specific to the field of *information systems* and to develop a well founded *procedure for specialising* this set, in a utility-driven and goal-bounded way, according to the requirements of specific situations pertaining to the development and evolution of information systems.

4 Utility of Concepts in Communication and Computation

Let us now develop a functional or *utilitarian* view on concepts as they are integrated in some *language* that is being used to communicate. This boils down to the question: 'what is it that concepts are *for*'?

4.1 A utilitarian view on concepts and meta-concepts

Roughly speaking, there are two main areas of use for concepts: *communication* and *computation*. Though these uses are often heavily entwined – to the point where they can hardly be distinguished – at a more fundamental level of analysis they are completely different [Hop03]. The distinction between communication- and computation-oriented concept use is related to the two general aspects discussed in section 3: the 'informational aspect' and the 'system aspect'.

Concepts for communication are bound up with languages in order to communicate: exchange information, and thereby ultimately change the knowledge of some individual in line with some intention of another individual to do so. Such concept use is very strongly tied up with communication between humans as studied in linguistics and communication theory. The utility of concepts for communicative use is therefore related to *principles of effective communication*.

Concepts for computation form *symbolic structures* that are usually intended as part of an engineered artefact. Even though their status is not primarily 'physical', there is essentially a clear and unambiguous link between the symbols in the structure (for example, programming code) and an underlying piece of hardware. This comes clearly to the fore in the case of assembly code, microcode and hardware, in particular in the trade-off between the realisation of computational functionality in hardware or microcode. The computational use of concepts is mostly tied up with fields like electronic engineering, computer engineering and software engineering. The utility of concepts for computational use is therefore related to *principles of good engineering*.

Ideally, a balance would need to be struck in each situation between the sorts of utility involved. Architects are perhaps the most important group of people that carry the burden of bridging the gaps between levels, layers, groups, and activities involved in information system development.

Both concepts for communication and concepts for computation are subject of discussion. Such 'meta-conversations' [Hop03] may concern the labelling of a concept (typically, the word form associated with it in some language), or the meaning of a concept. However, the very idea of (how to discuss and represent) 'meaning' is usually quite different in the two areas of use distinguished.

So as to avoid the discussion concerning "the meaning of meaning" [Put75], it is possible to take a strictly functional (i.e. *utilitarian*) approach to conversation about concepts. In that case, we look upon communication about concepts as striving for sharing knowledge about the meaning of a concept (according to whatever view on 'meaning' is deemed relevant) so that parties involved can, in their own opinion, effectively communicate.

The diversity of contexts for meta-conversation may differ quite substantially. In particular, the difference between meta-conversation in context of 'concept use for communication' will often be radically different from that in context of 'concept use for computation'.

4.2 Conversations for conceptual mediation

There is a third type of utility for concepts. In the various disciplines and activities involved in large-scale information system development, a multitude of terms, concepts, and languages is used, combining the two main utilities in many different ways. Straightforward strategies to deal with this are (assuming involvement of only two parties for sake of the argument):

- One party acquires/uses the vocabulary of the other.
- Both parties acquire/use each other's vocabulary.
- Both parties acquire/use a third vocabulary, a *lingua franca*.
- A third party steps in as a *translator*.

Note that these strategies apply to both types of utility in concept use, and also *between them*. However, the 'conceptual mediation utility of concepts' is especially closely related to meta-conversation. Strictly speaking, it may not only involve an active combination of *various* meta-languages and meta-conversation strategies, but even a class of meta-language/strategies of its own: meta-language specific to the task of bridging gaps between languages or (types of) meaning description. (For example, consider conversations between *translators*).

Architects, more than any other professional group, should be able to be active conceptual mediators. Modelling concepts, and the various meta-languages and conversation strategies that may be used to discuss them and align their meaning between parties, are in the center of the utilitarian (meta-)discussion because they are particularly vulnerable to confusion and incompatibility resulting from the communication-computation opposition, and related mix-ups.

5 Case studies

This section aims to illustrate the above discussed principles by briefly positioning two existing frameworks of viewpoints. In the discussions above, we have argued how the combination of a goal and a stakeholder concern should ideally dominate the choice of a specific meta-model when modelling a domain. The motivation for specific choices of concepts in the meta-model should be formulated in terms of the utility these concepts may bring towards the modelling goal and stakeholder concern. This utility may pertain to the potential communicative, computational as well as mediative use of the concepts.

In the case studies we will primarily focus on the viewpoints identified, the goals and concerns they aim to serve, the concepts in the associated meta-model, and any motivations for the specific concepts in the meta-model. The two case studies are not intended as criticism on the original frameworks. They may, however, indicate that some important considerations (such as utility based motivations) have been left out of the (original) publication of the frameworks. This does not imply that the framework as such is at fault, but rather that interesting and important motivations underlying the specific frameworks have been left implicit. Furthermore, the case studies are part of ongoing research efforts. A more detailed elaboration of the cases presented is part of these efforts.

5.1 The '4+1' view model

In [Kru95], Kruchten introduces a framework of viewpoints (a view model) comprising five viewpoints. The use of multiple viewpoints is motivated by the observation that it "*allows to address separately the concerns of the various stakeholders of the architecture: end-user, developers, systems engineers, project managers, etc., and to handle separately the functional and non-functional requirements*". Kruchten does not explicitly document the motivation for these specific five viewpoints. This also applies to the version of the framework as it appears in [Kru00, BRJ99].

The goals, stakeholders, concerns, and meta-model of the 4+1 framework can be presented, in brief, as follows:

Viewpoint:	Logical	Process	Development	Physical	Scenarios
Goal:	Capture the services which the system should provide	Capture concurrency and sychronisation aspects of the design	Describe static organisation of the software and its development	Describe mapping of software onto hardware, and its distribution	Provide a driver to discover key elements in design Validation and illustration
Stakeholders:	Architect End-users	Architect System designer Integrator	Architect Developer Manager	Architect System designer	Architect End-users Developer
Concerns:	Functionality	Performance Availability Fault tolerance ...	Organisation Re-use Portability ...	Scalability Performance Availability ...	Understandability
Meta-model:	Object-classes Associations Inheritance ...	Event Message Broadcast ...	Module Subsystem Layer ...	Processor Device Bandwidth ...	Objects-classes Events Steps ...

Note: in [Kru00, BRJ99], the viewpoints have been re-named; physical viewpoint → deployment viewpoint, development viewpoint → implementation viewpoint and scenario viewpoint → use-case viewpoint, to better match the terminology of the UML.

The framework proposes modelling concepts (the meta-model) for each of the specific viewpoints. It does so, however, without explicitly discussing how these modelling concepts indeed contribute towards the goals of the specific viewponts. Are, for example, object-classes, associations, etc, the right concepts to communicate with end-users about the services they required from the system? The 4+1 framework is based on experiences in practical settings by its author. This makes it even more interesting to make explicit the motivations, in terms of utility, for selecting the different modelling concepts. In [Kru00, BRJ99] this is also not documented. The viewpoints are merely presented 'as is'.

5.2 RM-ODP

The Reference Model of Open Distributed Processing (RM-ODP) [ISO98b, ISO96b, ISO96a, ISO98a] was produced in a joint effort by the international standard bodies ISO and ITU-T in order to develop a coordinating framework for the standardisation of open distributed processing. The resulting framework defines five viewpoints: *enterprise*, *information*, *computational*, *engineering* and *technology*. The modelling concepts used in each of these views are based on the object-oriented paradigm.

The goals, concerns, and associated meta-models of the viewpoints identified by the RM-ODP can be presented, in brief, as follows:

Viewpoint:	Enterprise	Information	Computational	Engineering	Technology
Goal:	Capture purpose, scope and policies of the system	Capture semantics of information and processing performed by the system	Express distribution of the system into interacting objects	Describe design of distribution oriented aspects of the system	Describe choice of technology used in the system
Concerns:	Organisational requirements and structure	Information and processing required	Distribution of system Functional decomposition	Distribution of the system, and mechanisms and functions needed	Hardware and software choices Compliancy to other views
Meta-model:	Objects Communities Permissions Obligations Contract ...	Object classes Associations Process ...	Objects Interfaces Interaction Activities ...	Objects Channels Node Capsule Cluster ...	Not stated explicitly

The RM-ODP provides a modelling language for each of the viewpoints identified. It furthermore states:

> *"Each language [for creating views/models conform a viewpoint] has sufficient expressive power to specify an ODP function, application or policy from the corresponding viewpoint."*

A more detailed discussion regarding the utility of the concepts underlying each of these languages, from the perspective of the goals/concerns that are addressed by each of the viewpoints

is not provided. The RM-ODP also does not explicitly associated viewpoints to a specific class of stakeholders. This is left implicit in the concerns which the viewpoints aim to address.

In particular in the case of an international standard, it would have been interesting to see explicit motivations, in terms of utility to the different goals, for the modelling concepts selected in each of the views.

6 Summary

Both of the discussed frameworks of viewpoints do not provide an explicit motivation for their choices regarding the modelling concepts used in specific viewpoints. When using one of the two frameworks, architects will not find it difficult to select a viewpoint for a given modelling task at hand. However, this 'ease of choice' is more a result of the limitation of the selections of options available (one is limited to the number of viewpoints provided by the framework) than the result of a well motivated choice about the viewpoint's utility towards the tasks at hand. Even more, making a choice for one of the two frameworks (or rather, one of the many, many, other frameworks) will be hard to make on rational grounds, without such utility based motivations.

7 Conclusion

The focus of this paper was on the principles underlying the utility of modelling concepts. We have discussed these issues from three angles: *modelling*, *utility* and *communication*. The primary area of application of the principles presented, has been the modelling of enterprise architectures and information system architectures, where the selection of concepts used to model different aspects very much depends on the specific concerns that need to be addressed. Rather than providing 'yet another framework of viewpoints', the aim of this paper was to lay a foundation that enables architects to *reason about* the many frameworks of viewpoints that are already available. We have illustrated our approach by a brief review of the relevant aspects of two existing frameworks for modelling (software intensive) information systems and their architectures.

Currently, further research activities are structured along three interlinked lines of activities. Firstly, the theoretical foundations touched upon in this paper are elaborated further. Results of this effort are expected to become part of the FRISCO revised report [VSP04]. Secondly, more frameworks of viewpoints will be studied and documented, and where possible, we will 'excavate' the underlying motivations for selecting modelling concepts in the viewpoints identified from the documentation available. Thirdly, interviews and workshops[3] will be organised with practising architects, with the aim of eliciting heuristics for deciding the potential utility of modelling concepts, meta-models and viewpoints. These three lines of activities are expected to influence each other in a synergistic fashion.

References

[BCK98] L. Bass, P. Clements, and R. Kazman. *Software Architecture in Practice*. Addison Wesley, Reading, Massachusetts, USA, 1998.

[BMS98] P. Bernus, K. Mertins, and G. Schmidt, editors. *Handbook on Architectures of Information Systems*. International Handbooks on Information Systems. Springer, Berlin, Germany, EU, 1998.

[Boa99] B.H. Boar. *Constructing Blueprints for Enterprise IT architectures*. Wiley, New York, New York, USA, 1999.

[3] These activities are part of the ArchiMate project (`http://archimate.telin.nl`), a research initiative that aims to provide concepts and techniques to support enterprise architects in the visualisation, communication and analysis of integrated architectures. The ArchiMate consortium consists of ABN AMRO, Stichting Pensioenfonds ABP, the Dutch Tax and Customs Administration, Ordina, Telematica Instituut, Centrum voor Wiskunde en Informatica, Katholieke Universiteit Nijmegen, and the Leiden Institute of Advanced Computer Science.

[BRJ99] G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modelling Language User Guide*. Addison-Wesley, Reading, Massachusetts, USA, 1999.

[FHL+98] E.D. Falkenberg, W. Hesse, P. Lindgreen, B.E. Nilsson, J.L.H. Oei, C. Rolland, R.K. Stamper, F.J.M. Van Assche, A.A. Verrijn-Stuart, and K. Voss, editors. *A Framework of Information Systems Concepts*. IFIP WG 8.1 Task Group FRISCO, 1998.

[Hop03] S.J.B.A. Hoppenbrouwers. *Freezing Language; Conceptualisation processes in ICT supported organisations*. PhD thesis, University of Nijmegen, Nijmegen, The Netherlands, EU, 2003.

[IEE00] Recommended Practice for Architectural Description of Software Intensive Systems. Technical Report IEEE P1471-2000, IEEE Standards Department, The Architecture Working Group of the Software Engineering Committee, September 2000.

[ISO96a] *Information technology – Open Distributed Processing – Reference model: Architecture*, 1996. ISO/IEC 10746-3:1996(E).

[ISO96b] *Information technology – Open Distributed Processing – Reference model: Foundations*, 1996. ISO/IEC 10746-2:1996(E).

[ISO98a] *Information technology – Open Distributed Processing – Reference model: Architectural semantics*, 1998. ISO/IEC 10746-4:1998(E).

[ISO98b] *Information technology – Open Distributed Processing – Reference model: Overview*, 1998. ISO/IEC 10746-1:1998(E).

[JBA+03] H. Jonkers, R. van Buuren, F. Arbab, F. de Boer, M. Bonsangue, H. Bosma, H. ter Doest, L. Groenewegen, J. Guillen Scholten, S.J.B.A. Hoppenbrouwers, M.-E. Iacob, W. Janssen, M.M. Lankhorst, D. van Leeuwen, H.A. Proper, A. Stam, L. van der Torre, and G.E. Veldhuijzen van Zanten. Towards a Language for Coherent Enterprise Architecture Descriptions. In M. Steen and B.R. Bryant, editors, *7th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2003)*, pages 28–39, Brisbane, Australia, September 2003. IEEE Computer Society Press, Los Alamitos, California, USA.

[Kru95] P. Kruchten. The 4+1 view model of architecture. *IEEE Software*, 12(6):42–50, November 1995.

[Kru00] P. Kruchten. *The Rational Unified Process: An Introduction*. Addison-Wesley, Reading, Massachusetts, USA, 2nd edition, 2000.

[KZR04] R. Kishore, H. Zhang, and R. Ramesh. A helix-spindle model for ontological engineering. *Communications of the ACM*, 47(2):69–75, 2004.

[Pei69a] C.S. Peirce. *Volumes I and II – Principles of Philosophy and Elements of Logic*. Collected Papers of C.S. Peirce. Harvard University Press, Boston, Massachusetts, USA, 1969.

[Pei69b] C.S. Peirce. *Volumes III and IV – Exact Logic and The Simplest Mathematics*. Collected Papers of C.S. Peirce. Harvard University Press, Boston, Massachusetts, USA, 1969.

[Pei69c] C.S. Peirce. *Volumes V and VI – Pragmatism and Pragmaticism and Scientific Metaphysics*. Collected Papers of C.S. Peirce. Harvard University Press, Boston, Massachusetts, USA, 1969.

[Pei69d] C.S. Peirce. *Volumes VI and VIII – Science and Philosophy and Reviews, Correspondence and Bibliography*. Collected Papers of C.S. Peirce. Harvard University Press, Boston, Massachusetts, USA, 1969.

[Put75] H. Putnam. The meaning of meaning. In *Mind, Language, and Reality*, Cambridge, Massachusetts, USA, 1975. Cambridge University Press.

[The04] The Open Group. *TOGAF – The Open Group Architectural Framework*, 2004.

[VSP04] A.A. Verrijn-Stuart and H.A. Proper, editors. *Framework of Information Systems Concepts: Revised Report*. IFIP WG 8.1 Task Group FRISCO, 2004.

[WAA85] A.T. Wood-Harper, L. Antill, and D.E. Avison. *Information Systems Definition: The Multiview Approach*. Blackwell Scientific Publications, Oxford, United Kingdom, EU, 1985.

[Zac87] J.A. Zachman. A framework for information systems architecture. *IBM Systems Journal*, 26(3), 1987.