Views in the Enterprise Domain

Richard Martin[1] and Edward Robertson[⋆2]

[1] Tinwisle Corp., Bloomington IN 47404, USA, `tinwisle@bloomington.in.us`
[2] Computer Science Dept., School of Informatics, Indiana Univ., Bloomington IN 47405, USA, `edrbtsn@indiana.edu`

Abstract. This paper explores the notions of view and related concepts as they are used in high-level modeling, particularly in certain framework standards of the International Standards Organization. The purpose is to highlight the similarities and differences between usage in this realm and the realm of implementation.

1 Introduction

There are many views of "views", even within the context of computer systems. As is the general case where there are multiple views, this indicates a healthy variety of perspectives. As is also the case, a useful synthesis of these views requires understanding the perspectives involved. This paper discusses some of these perspectives a goal of achieving a broader understanding of views.

Another goal of this paper is to mediate between the concepts and terminology of the enterprise standards world and the world of object oriented programming and modeling. A significant example of this, elaborated later, concerns the meaning of "object view", which is much more specific in certain standards than the common interpretation. Note that we do not aspire to a common definition of "view"; in fact, we believe that a single monolithic definition is not desirable, much less achievable. Rather, we seek to align the various notions so that their points of contact and points of difference are both apparent.

In particular, we hope to explicitly characterize all facets of views and viewing in a scope comprising enterprise architectures and enterprise architecture standards. Because of the wide range of people and activities within this scope, we hope to accommodate a wide range of views and view uses. Longer term objectives include facilitating use of views in design and developing view formalisms suitable for application and implementation.

Having broached the notion of perspective, it behooves us to be explicit about ours. Although software professionals, we have become involved with specification and modeling at the highest organizational levels, where software is only one component (although facilitating computer-based information management and exchange is a common goal of this modeling). One facet of this involvement

⋆ Supported by NSF IDM 82407

is with international standards intended to facilitate interoperation of manufacturing and related enterprises. In particular, Richard Martin is convener of ISO TC184/SC5/WG1.[3]

An enterprise architecture framework is a means to understand an enterprise or class of enterprises by organizing and presenting artifacts that conceptualize and describe the enterprise. The canonical framework is that proposed by Zachman.[9] An enterprise is a collective activity in a particular domain with actors sharing a common purpose; an enterprise can be a business, a collection of businesses along a supply chain, a government agency, *etc.* Architecture is a metaphor from the realm of office towers and bridges, intended to capture the use-oriented, as opposed to construction-oriented, aspects of the design of those structures. A framework is a structured container for holding and interconnecting things, in this case model artifacts.

For several decades, committees within the ISO have documented frameworks as a means to standardize formulating and managing the many artifacts created during systems development and operation. These committees have adopted a range of perspectives to provide organizational focus for these frameworks.

The standards on which we focus are those associated with ISO TC184/SC5 and other committees and organizations with which TC184/SC5 has liaison. While each model artifact has a unique place in a Zachman frame, these artifacts occur redundantly in the standards, highlighting that views are necessary for these standards. A table in the appendix lists several relevant standards. Throughout this paper, "standards" refer to those standards, as these are the ones with which we are most familiar and in whose development we have participated.

While the scope of these standards is nominally far beyond computer systems, in reality they deal heavily with information systems. At the enterprise level, the computer systems that matter most are information systems, where the technological foundations of these information systems are means, not ends. Even though the architectures describe enterprises far beyond computer systems, the architectures are information artifacts, typically with computer-based representations.

In addition to accommodating multiple perspectives, views are used in standards to: examine and define content, expose content to enable interoperability, reduce apparent complexity, provide focus, enable modularity of process, and enforce "need to know" restrictions.

This paper is organized as follows. The next section distinguishes meanings given to the term "view" in various contexts. The following section relates views and meta-levels, in recognition of the difficulties that arise when meta-levels

[3] International Standards Organization (ISO) Technical Committee 184 (Industrial automation systems and integration), Sub-Committee 5 (Architecture, communications and integration frameworks), Working Group 1 (Modeling and architecture). Also, within ISO "DIS" abbreviates "Draft International Standard" and "FDIS" abbreviates "Final Draft International Standard". Adopted standards only have the "ISO" designation.

are incorrectly considered when dealing with views of all kinds. Having set the ground, Sect. 4 discusses specific standards with respect to views and the approaches employed to manage those views. The always thorny issue of views and updates is discussed in the next section. The final section summarizes and looks forward.

2 Distinctions: the Many Views of "View"

Distinguishing the many meanings of "view" and differentiating those meanings from other words commonly thought to be close to "view" is necessary before delving into the details of views. We have discovered, particularly in the context of developing standards, that there are significant mismatches in understanding "view" when dealing with multiple constituencies. This problem is particularly surprising when the parties are all well versed in analysis and modeling. Developing international standards brings the additional problem of translations – a terminological problem is exacerbated when multiple meanings translate to different words or when several words translate to one word with distinct meanings.

The first distinction is that "view" is both a verb and a noun. While it is certain that one meaning of "to view" is "to create a view", there are other situations in which a view is at best implicitly and weakly constructed by viewing. In any case, a view is always associated with a perspective or *viewpoint*. A view is something observed while a viewpoint is where the observer stands.

This verb/noun distinction is particularly useful in differentiating "view" from another verb/noun word: "model". Both a view and a model are artifacts and it is conceivable that a model and a view could be identical. However, the processes of modeling (synthesis) and viewing (analysis) are quite different; A view is created by different means and with different goals than a model. In other words, a view and a model might be the same in extension but are always distinct in intension.

A view is (ideally) an image, often intended to capture some particular perspective. A model is constructed or synthesized, often from components capturing several different perspectives. Moreover, a model is often characterized by a particular *medium*, whether that medium is plastic (as in model airplanes) or notational (as in ER models).

A view has a single viewpoint while a model may have many viewpoints, as many as there are observers of a model. Indeed, the range of a model may be assessed by the number of views that it supports.

Having distinguished meanings of "view" and "model", we now recognize that a common usage of "view" within the standards confounds these meanings, since a view is a constructed artifact (as with a model) but from a particular viewpoint.

A significant and beneficial shift in enterprise standards definitions is the recognition that viewpoints are as significant as views, that is, specifying concerns addressed by a view is as important as merely specifying its contents. In particular, a *user view* is a description of content while a *modeler view* is a statement of perspective. Just as a view from an observer on the hill looses acuity at the periphery of the visual field, a modeler view tends toward lower definition away from the center of content. User views, by contrast, almost always are well-defined throughout.

Historically, the standards have mandated a fixed set of views, which we call *prescribed* views. Typical prescribed views concern information and function (thus corresponding to parts of the "interrogative" dimension of Zachman's framework). Such prescribed views are often associated with fundamental roles in model generation. Other prescribed views are domain specific, associated with a fixed concern for the particular enterprise at hand (*e.g.* the network view of a communications provider). In the adopted standards, all such prescribed views are modeler views. Recently, in drafting new standards, there has been a growing realization that a more general and flexible viewing mechanism must exist, allowing for extraction, rearrangement, and insertion of content by both modelers and users. Thus any view mechanism must provide for *ad hoc* as well as prescribed views.

A viewpoint is often associated with one or more human roles, in particular the concerns and responsibilities of those roles. The correspondence is not one to one, however, in that a role (particularly a role that bridges domains) may have more than one view and of course one view may be shared by several roles. The user/modeler distinction also applies to roles. All of our enterprise framework and standards activities concern modeler roles. Indeed, three broad categories of (modeler) role characterize one axis of Zachman frameworks, namely: *specifier*, *designer*, and *builder*. User roles (operational personnel) appear within the cells of a framework. Thus there seems to be no point of contact with the OO interpretation of "Role".

Every organizing characteristic can be the basis for a view, but it is inadequate to think of these characteristics as merely views. This is particularly true when characteristics provide a coherent dimension, such as the Zachman roles.

Finally, we itemize a number of ways in which views are used in the standards and discuss a number of ancillary issues. Views are used:

- as means for encouraging consistent and complete analysis and modeling
- as an aid to user understanding
- as a means for expressing characteristics of interoperability
- as an expression of model content
- as a means to add new content

Views are not used to their full potential in standards activities (and elsewhere). Views are not currently considered as a mechanism for verifying consistency, completeness, and interoperability. However, there is nascent activity in that area. There is an informal appreciation that the content of a model may need to be controlled in order to satisfy a particular view.

3 Views and Meta Levels

The preceding distinctions on "view" are obviously impacted by meta-levels. A modeler operates at a different abstraction level than a user. But view also impacts meta considerations. Thus it is worthwhile to examine more closely the connections between views and levels of abstraction (meta-levels).

First, we observe that "meta" is a relative, not an absolute, characterization. This is evidenced by the fact that there are contexts where the locution "meta-meta-data" is meaningful. As an example of a many-tiered meta hierarchy, the Table 1 illustrates a spectrum of "metas" in information space, working down from the most abstract to the real world (note there are two paths down the hierarchy which coalesce toward the bottom, an intriguing characterization of subtypedness). Each level has one or more examples of the "things" that are the direct content of that level and the corresponding meta notion. Observe that a construct at one level is a meta-construct at the next level down. OMG's Meta Object Facility (MOF)[6] displays a similar structure, although the top level is mechanism and not concept.

	meta-thing	**thing**	**example of view**
Concept Level	entity	Entity (type)	temporal concepts
	subtype	IsA	
Model Level	Entity (type)	`Person`	Information View (14439)
	IsA	`Employee` as subtype	
Data Level	`Person`	entry for Joe	something defined using an RDB `VIEW`
	`Employee`		
World Level	entry for Joe	the real Joe	a view of the Firth of Clyde

Table 1. Examples of Meta Hierarchy and Views

The example hierarchy in Table 1 is specifically a hierarchy of modeling and abstraction, but there are other uses of meta as well, even in the information space. Much scientific information is classified as either data (such as the actual results of experiments) or metadata (such as the dates and conditions of experiments). In Relational DataBases (RDBs), the metadata is structural; that is, RDB metadata is the table and attribute names in the relational schema. More broadly, "meta" always indicate a hierarchy, but the nature of the hierarchy may depend upon context.

Understanding the meta hierarchy helps to eliminate confusion, such as when `IsA` ambiguously covers subtype and membership.

Views exist at all meta levels, as illustrated in the last column of Table 1. However, they are most common and significant at the data and model levels. Views at the data level are commonly user views, while (unsurprisingly) modeler views dominate the model level. Views may have effect up or down the meta hierarchy. A view of structure is meta (*i.e.* up-hierarchy) with respect to the data, as illustrated in the RDB context in the example. On the other hand, view definition is often done at a higher meta-level and propagated down the hierarchy. (*e.g.*, financial views become accounting practices).

To achieve a consistent context within a multi-level view, it may be necessary to create missing abstract meta-information or make concrete the data associated with a meta element.

Process views introduce yet another kind of context – that of decomposition or level of detail. Note that, as we argue in [5], abstract ↔ concrete and coarse ↔ fine detail are distinct scales.[4] For example, in one view `assemble activity` is a single step while in another at the same level of abstraction, `assemble` is a complex process with many subsidiary activities. Business level processes do not well-represent operational processes, such as shop-floor activities whose scheduling and resource dependencies are absolute.

4 Views in Specific Standards

While views play only a limited role in completed (*i.e.* adopted) standards, there is, within standards bodies, a growing recognition of the importance of views. Views in adopted standards are almost all prescriptive views, insuring that certain viewpoints/roles are fully involved in model development. On the other hand, ongoing standards development is cognizant of the distinctions discussed above. This section is thus written with the context of standards development.

In the previous discussion of views, we characterized views and the viewing issues at a level of abstraction beyond implementation. While the query mechanism of databases can create complex views of information, the remaining distinguished concepts of view lack such a mechanism. On the other hand, the notions of view and viewpoint act as drivers during the modeling of new capabilities, enhancing our understanding of the many ways in which model elements can be expressed. We wish to focus on the characteristics of any view rather than those of a particular view, such as function view or specifier role. We now draw attention to the implementation side of view as it applies to enterprise modeling.

Some of the practical consequences of our conceptual understanding of views as they relate to modeling are beyond technical challenges. The distinction between modeler view and user view is particularly vexing in practice. This is partially because the modelers are in fact users.

Yet the work of ISO TC184/SC5/WG1 on modeling and views provides no general principles to guide the mapping of modeling constructs, specified in ISO/DIS 19440, into views of the model, specified in ISO/FDIS 19439. ISO/FDIS 19439 articulates a framework for enterprise modeling in the realm of industrial automation. This framework is constructed along three dimensions: View, Model Phase, and Genericity. The View dimension is the canonical source of prescribed views. The process suggested in the 19439 draft begins by developing the function view, from which information and resource views are extracted. Observe

[4] Furthermore, a third scale, scope, ranging from generic to specific, is independent of the other two. This scale is often used in characterizing standards.

that the intended process involves construction through views. ISO/FDIS 19440 enumerates a set of constructs that are intended to be building blocks for model artifacts found populating the cells of the three dimensional space characterized by 19439. While the modeling standard (19439) contends with the distinction between verb and noun, the construct standard (19440) uses the noun "view" to characterize a representational form needed for implementation of the manufacturing process.

ISO/DIS 19440 calls this representation an *Object View* (OV), a notion which is derived from the corresponding model element of the CIMOSA Baseline[2]. An Object View has nothing to do with the prescriptive views of ISO/FDIS 19439, but is characterized in 19440 as a representation of an *enterprise object* (EO) that "enable[s] (i) the capturing of relevant information needed for strategic planning and decision support, and monitoring and control of operational processes and (ii) its presentation in both human and machine understandable form." An EO may not be represented in medium suitable for an intended use; in this case, an OV is created in the appropriate representation. An OV, according to the standard, is a modeling/design artifact, but it is best illustrated at the instance level. For example, a Shipping Order (the EO) is projected down to a printed Pick List (the OV), so that shipping personnel can gather the products to fulfill an order.

As in the above example, an OV is always transitory. Thus the OV construct can be seen as a practical mechanism for casting an endurant object into a perdurant object that need not persist beyond its usage context,[5] such as a process input that is only relevant during the process execution.[8] OV's share this temporal flavor with OO Roles, but there is a subtle difference in intent. Management of OV's using OO Role techniques should certainly be considered by the standards bodies.

An OV is never itself an object. In particular, it is not possible to create an OV of an OV, unlike many other situations where "view chaining" is useful.

While the OV construct provides a basis for viewing a single object in a standard way, such a standard for a view comprising several objects remains an open problem currently resolved by *a priori* assignment of view inclusion through "object view" template instantiation. We note with interest a proposal by David Shorter[7] to express OV representations using the familiar Observer pattern.[3] Perhaps this path will lead to an effective resolution of the *ad hoc* multi-object view implementation problem.

To produce views without relaxing constraints on objects we must accommodate both the encapsulation barrier surrounding an object's core and retain necessary dependency relationships. These accommodations are embodied in programmatic artifacts and required normalization rules for data structures.

[5] Endurant things are instantaneous "snapshots" of perdurant things. Another characterization places perdurant things in a 4-D world (space and time) while endurant things are in some 3-D world (only space).

5 Views and Updates

Update *via* views is always a thorny issue. The thorns are particularly sharp when the updates cross meta-levels. For example, occasionally a characteristic (*e.g.* temperature) is added to an operational component and this addition must be propagated to all components of the same type. In the RDB world, where `VIEW` has a precise meaning, the problem of updating *via* views has been well-studied[1, 4] (but is still problematic).

One difficulty with the Object View, like any temporary object, is that constraints on the Enterprise Object from which it is cast must be maintained and constraints modified for the particular Object View instance use must not conflict with that EO. Then, as the OV is used, possible conflicts between the OV and its EO must be communicated in some fashion. Design patterns,[3] or at least the conceptual notions they embody, can be used as possible mechanisms to aid in this communication. An Observer pattern[3] is a simple means to provide consistency in this case, but the Interested Party pattern (Shorter's[7] elaboration of the Observer pattern, as suggested in [3]) would provide additional consistency management features. The Interested Party pattern adds a change manager to register what modifications each view needs to be notified about.

Not as clear are patterns that propagate changes from the OV back to its EO. This parallels the familiar view update problem of database systems, exacerbated by the fact that a single EO may have several OV's, each expressing a distinct role of the EO. The current ISO/DIS 19440 avoids the problem by restricting OV use to inputs and outputs in situations where updates to the EO are not relevant. The OV is created when needed and discarded when the need no longer exists.

For operational systems, views help to partition and organize enterprise artifacts but are not maintainers of particular artifacts. Views can be used to examine existence and dependency relationships. User views are most often analytical representations of models in a domain that observe change but do not produce change.

The heavy use of well defined views of relational and object-oriented database products attest to the importance of supporting users' expectations for selective information access. This domain has also characterized the pitfalls of extending view use for information maintenance. Recent work by J. Lecthenbörger[4] stresses the importance of restricting view generated updates to those which are reversible within the view and result in no change to the complement of the view.

Another aspect of views and updates is how views reflect changes in the base information. This is well-studied in RBDs as the "view maintenance problem." Unfortunately, current standards do not adequately address the issues involved. However, the Interested Party pattern does provide a solution.

6 Looking Back, Looking Forward

Today's *force majeure*, XML, is going to have a profound impact on view use precisely because its manifestations are becoming the communication channel and thus its characteristics limit the channel. This is in fact just the next phase of a continuing trend – that technology drives (and often restricts) conceptualization. Maximizing the semantic qualities of a view transmitted along that channel poses even more challenges. There are technical issues as well: XPATH expressions given in XQuery statements result in lists or tree transformations. The ordering within an XML tree imparts a path to the objects contained on the tree. Such an order can be disrupted during transformations intended to render a particular view, resulting in a distortion.

The other side of view, that which seeks to influence a model's construction and use, is often manifest in meta-models intended to guide system development. The view dimension of ISO/FDIS 19439 organizes model artifacts into categories of function, information, resource, and organization. For each category of the "view" aspect, model artifacts are placed in phases corresponding to use throughout the life-cycle of the enterprise. These phases constitute the "model phase" dimension, which has a temporal flavor. ISO 15704 further expresses the importance of view in meta-model creation with additional views for the economic and the decision aspects of the enterprise. These views correspond to the observations from different viewpoints. Both of these additional views express new aspects of dimensionality – the economic view includes three levels of content detail (economic targets and constraints, influencing factors, and base elements) and the decision view levels are grouped by scope and time. Realistically, a viewpoint is more likely to be a space of multiple observations distilled into a selective perception of some process or end state.

Therefore, views both guide model development and support the operational functions resulting from model use. The effort of the OMG in assisting the emergence of SysML extensions to UML for system engineering support proposes a "system view" class to facilitate the collection of model elements into view representations. Perhaps the simplest expression of viewpoint as a driver for system development is found in IEEE-1471 where the relationships among stakeholders, concerns, viewpoint, view and model is presented in the context of architectural descriptions.

There is a continuing struggle to produce interfaces between human and machine that allow a selective view to addresses a particular perception. Tools must be developed to facilitate an even higher-order interactive level of monitoring and managing model artifacts. Because we build models to support views of a user, as we build those models we are constantly estimating the required support during design. A better understanding of the impact user's views have on model use is driving improvement in model development activities.

Views are the mechanism by which we conceive our models and observe the behavior resulting from that model's use.

References

[1] F. Bancilhon and N. Spyratos. Update semantics of relational views. *ACM Trans. on Database Systems*, 6:557–575, 1981.

[2] CIMOSA Association. Open System Architecture for CIM, technical baseline; version 3.0, 1994.

[3] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.

[4] Jens Lechtenbörger. The impact of the constant complement approach towards view updating. In *Proc. PODS 03*, pages 49–55.

[5] Richard A. Martin, Edward L. Robertson, and John A. Springer. Architectural principles for enterprise frameworks. In Janis Grundspenkis and Marite Kirikova, editors, *Knowledge and Model Driven Information Systems Engineering*, pages 151–162. Riga Technical University, 2004.

[6] Object Managment Group. Meta Object Facility specification. Technical report, 2000. available at `www.omg.org`.

[7] David Shorter. Integrating enterprise model views through alignment of metamodels. In *Proc. 2004 Int. Conf on Enterprise Integration and Modelling Technology.*

[8] John G. Stell and Matthew West. A four-dimensionalist mereotopology. In A. C. Varzi and L. Vieu, editors, *Formal Ontology in Information Systems*, pages 261–272. IOS Press, 2004.

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

A Appendix

Standards particularly relevant to this paper:

standard	**name**	**status**	**responsible**
ISO 10746	Open Distributed Processing	appr. 1996, 1998	JTC1 ISO/IEC
ISO 14258	Concepts and Rules for Enterprise Models	appr. 1998	TC184/SC5
ISO 15704	Industrial automation systems – Requirements for enterprise-reference architectures and methodologies	appr. 2000	TC184/SC5
ISO 19439	Enterprise Integration - Framework for Enterprise Modeling	final draft	TC184/SC5
ISO 19440	Enterprise Integration - Constructs for Enterprise Modeling	draft	TC184/SC5
IEEE 1471	Recommended Practice for Architectural Description for Software-Intensive Systems	appr. 2000	ANSI/IEEE
ENV 40 003	Computer Integrated Manufacturing – Systems Architecture - Framework for Enterprise Modelling	appr. 1990	CEN

All ISO standards are available from `www.iso.org`. CEN is the European Committee for Standardization (*Comité Européen de Normalisation*, `www.cenorm.be`).