3. Integrating Enterprise Model Views through Alignment of Metamodels

David Shorter, IT Focus
Convenor, CEN TC310 WG1[6]
david.itfocus@zen.co.uk

Standards have been developed and are still developing for enterprise modelling frameworks and modelling constructs. Recently they have made increasing although informal use of UML to metamodel and hence to clarify the concepts involved. However, the relationships between those concepts (for example, positioning the modelling constructs within a framework) have not been defined to the degree of precision required. This paper describes an approach as a work-in-progress, which proposes to use metamodelling to ground the concepts within the framework, and so to resolve difficult issues such as federated views on an enterprise model. In the longer term it should also provide benefits through alignment with Object Management Group (OMG) developments, especially the Model Driven Architecture, MDA™.

1. INTRODUCTION

Two major strands can be identified in standards-related work on enterprise integration. They are:

- architectures and conceptual frameworks for the design of an enterprise, and
- concepts from which to develop enterprise models as the basis of enterprise integration.

In the manufacturing domain (although also applicable more widely), the following standards are currently available or due in the near future:

- a general framework (IS 15704)[7] for the assessment of methodologies and reference architectures,
- a more specific conceptual framework for enterprise integration (ENV 40003), which focuses on model-based integration – this standard is being superseded by (prEN/FDIS 19439), which has been aligned with the (IS 15704) framework, and
- constructs for enterprise modelling (ENV 12204), which is being superseded by (prEN/DIS 19440).

(prEN/DIS 19440) defines a modelling language *construct* as an "element of a modelling language designed to represent in a structured way the various information about common properties of a collection of enterprise entities". This definition specialises but aligns with that of (ISO 10303) which defines a construct as "a generic object class or template structure, which models a basic concept independently of its use". Given this 'object-orientation', the drafting process for

[6] Following a meeting of ISO TC184 SC5 WG1 (12-14/10/2004), this paper incorporates revisions to clarify the structure and purpose of the metamodel for [prEN/DIS 19440].
[7] References are listed in Section 8.

(prEN/DIS 19440) naturally made extensive use of the class notation of the Unified Modelling Language™, UML™ [8], to represent constructs and the relationships between them within an integrated metamodel.

The reasons for a model-based approach to specifying the constructs are well-argued in (Flater 2002): "But as the emphasis has shifted increasingly towards conceptual design, there has been an increasing role for specifications that deal with 'meta-level' concepts, or that relate to the mapping from these concepts down to the implementation level. These specifications do not fit naturally within the requirements of standards for software component interfaces ... yet the need for them is undisputed." Additionally "UML models capture more information at the concept level and more of the intent of the designer. This makes them more effective tools of communication to assist (standards developers[9]) in understanding each other and to assist users of standards in understanding the intent".

UML has achieved wide industry acceptance but is still evolving. The version used for the (prEN/DIS 19440) work was UML 1.4. The current specification is UML 1.5, which has added action semantics – not relevant for class modelling. However, UML is evolving to UML v2.0, which will have a more rigorous specification (achieved by metamodelling UML itself, see below) and which will integrate with the Model Driven Architecture, MDA™ 10 to provide platform independence and to enable composable models.

The (MOF 2002) specification explains that "the UML and MOF are based on a conceptual layered metamodel architecture, where elements in a given conceptual layer describe elements in the next layer down. For example,

- the MOF metametamodel is the language used to define the UML metamodel,
- the UML metamodel is the language used to define UML models, and
- a UML model is a language that defines aspects of a ... system[11].

"Thus, the UML metamodel can be described an 'instance-of' the MOF metametamodel, and a UML model can be described as an 'instance-of' the UML metamodel. However, these entities need not necessarily exist in the same domain of representation types and values. This approach is sometimes referred to as *loose metamodelling*."

Using UML in developing (prEN/DIS 19440) has been helpful – however the integrated metamodel of the constructs is not a normative part of the standard, and there is no publicly available version[12]. Also the way in which the Constructs[13] are to be deployed within the Framework is described only in illustrative terms. As

---

[8] http://www.uml.org/

[9] Author's insertion

[10] http://www.omg.org/mda/

[11] The [MOF 2002] specification says "of a computer system" but that is seen as over-restrictive.

[12] This metamodel uses UML notation but – for reasons of consistency checking with construct templates – contains several relational attributes which would normally be regarded as redundant since the relationships between constructs are shown explicitly.

[13] Hereafter 'Constructs' will be used as an abbreviation for "constructs as defined in [prEN/DIS 19440]", and 'Framework' for "[prEN/FDIS 19439] Framework"

explained in the next section, this is a particular problem for *views* on a model composed of *Constructs*.

This paper will argue that it is possible and desirable to:

- use metamodelling as a basis for reasoning about problematic issues such as views on an enterprise model,
- represent the concepts of the (prEN/FDIS 19439) framework in a metamodel,
- provide a 'mapping' mechanism, called **F**ramework-**C**onstruct-**M**apping or **FCM**, which will allow standardizers and modellers to refine relationships between the Framework and the Constructs, and
- base both the framework metamodel and the FCM on the UML meta-language (as is already largely the case for (prEN/DIS 19440)), and so enable integration within the MDA.

2. THE ISSUE OF VIEWS

A common theme for the model-related content of the Framework and Architecture standards (ENV 40003), (IS 15704), (prEN/FDIS 19439), and other frameworks such as the Zachman Framework for Enterprise Architecture[14] is the notion of *views* on a model. In (prEN/FDIS 19439), Enterprise Model Views are defined as "a selective perception or representation of an enterprise model that emphasizes some particular aspect and disregards others". This definition is well-aligned with the (OMG MDA Guide 2003) definition of *view* as "a viewpoint model or view of a system (that) is a representation of that system from the perspective of a chosen viewpoint", where *viewpoint* is defined as "a technique for abstraction using a selected set of architectural concepts and structuring rules, in order to focus on particular concerns within that system".

The motivation for such views is that in any real-world complex enterprise, there are multiple concerns to be modelled (e.g. functional, informational, economic, decisional) involving a combination of concern-specific and more general concepts, and that views allow the modeller and model-user to concentrate on only those details that are relevant for the purpose. However, the reconciliation of independently developed concern-specific models would be a problematic, if not impossible, exercise. The approach adopted in (prEN/FDIS 19439) is therefore to postulate an 'integrated enterprise model'[15], and to regard these concern-specific models (Enterprise Model Views) as representing major aspects of that integrated model and corresponding to a view on the enterprise itself. This corresponds to the 'federated development' approach (Whitman *et al* 1998) as described later and is also analogous to the 'Single Model Principle' (Paige and Ostroff 2002) which has been proposed to address consistency of UML modelling and the UML itself.

Just what the minimum set of views is to be, or whether there should be a minimum set, has been a matter of some debate. In an early standard (ISO 14258) two such views were defined – Function and Information. (ENV 40003 and IS 15704) later extended these to Function, Information, Resource and Organisation, but the

[14] http://www.zifa.com/

[15] [IS 15704] distinguishes between enterprise-reference architectures and methodologies that are model-based, and those that are not.

possibility of other views being required was accepted. More recently (prEN/DIS 19440) has been drafted to include a construct (Decision Centre) to support a Decision View, and it is anticipated that a future revision of (ISO 14258) will also include an Economic View.

Model consistency requires that changes made in any one view be propagated to the integrated model, and to any other view that is affected by that change. The issue is discussed in (Whitman *et al* 1998) which distinguishes between:

- a master view, requiring all relevant information to be entered in a single view ("found by several to be difficult if not impossible");
- the driving approach, where a view with the largest content is populated first (e.g. the Function View for CIMOSA) and then other views are populated from that information; and
- the federated approach "which allows the user to populate each view as information becomes available in that view (with the advantage of allowing) the addition of knowledge in the view most conducive to that form of knowledge".

However, this requires that whichever "approach (is used, it needs to) ensure the consistency between views" (*ibid*). The authors also point out that "this method is highly tool-dependent' and "the rigor of the tool capability in ensuring the proper mapping between views is critical to the success of this method..."

A central issue with the *views on an integrated model* approach is therefore how to guarantee that views (and changes made to the content of a view) are consistent. (prEN/DIS 19440) partly addresses this by the principle of representing the integrated model for any specific enterprise in terms of constructs which are themselves described in an integrated metamodel. However, other mechanisms are also needed to manage changes in the model content.

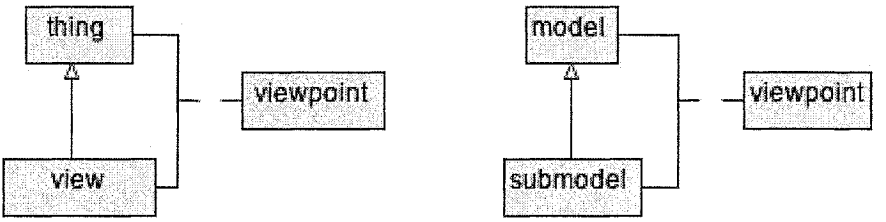

Figure 1 – Views and viewpoints

At its most general, a view is a selective perception from a particular viewpoint, so it is a selector of something – a mapping from a thing to a view on that thing, e.g. from a model to a submodel. (See Figure 1.)

A view is also a selective encapsulation of a modelled object's attributes or state (or those of a collection of modelled objects). It involves selection corresponding to the viewpoint, and encapsulation because it identifies and contains the information (state) relevant to that selected view.

This is not the same as the (Gamma *et al* 1997) *State* pattern because that externalises the state as an abstract class, which has alternative subclass implementations. Views are not alternatives, because several views can exist

simultaneously of the same modelled object(s) and some object attributes may be present in different views.

However a view is valid only if the content of the view is not changed independently of the modelled object(s) visible in the view. Views are therefore more akin to an *Observer* pattern (or *Model-View-Controller*), because any change to the content of a view has to cause a corresponding change in the state of the enterprise object or enterprise objects being viewed.

An implementation model[16], adapting the (Gamma *et al* 1997) Observer pattern, might appear as Figure 2[17].

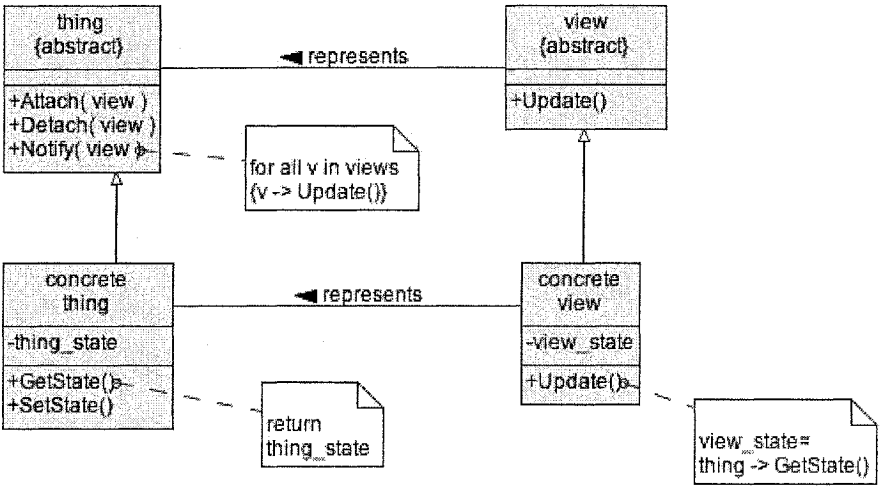

Figure 2 – Implementation model for views (*observer* pattern)

Something additional is needed to record each view's subset of the modelled object(s) state, for example as in Figure 3, modifying the approach to include aspects of interest as suggested by (Gamma *et al* 1997) in "specifying modifications to interested parties explicitly".

One issue is what is to be notified from the changed object to the interested party, because this will limit the extent of unification of the underlying model. For example, propagating attribute changes seems straightforward, but what of propagating, say, constraints on a modelled object? Or specialisations thereof?

3. LIMITATIONS OF ENTERPRISE OBJECTS AND OBJECT VIEWS IN (PREN/DIS 19440)

In (prEN/DIS 19440), an Enterprise Object (EO) is defined as "a construct that represents a piece of information in the domain of the enterprise (and) that describes

[16] Which might support for example a software tool to present view on models generated from prEN/DIS 19440 constructs

[17] The terms *thing* and *view* have been chosen to avoid confusion with (Enterprise) Object and Object View – the latter having a distinct and restricted meaning as described in the following section.

a generalized or a real or an abstract entity, which can be conceptualized as being a whole", and an Object View (OV) as "a construct that represents a collection of attributes of an Enterprise Object".

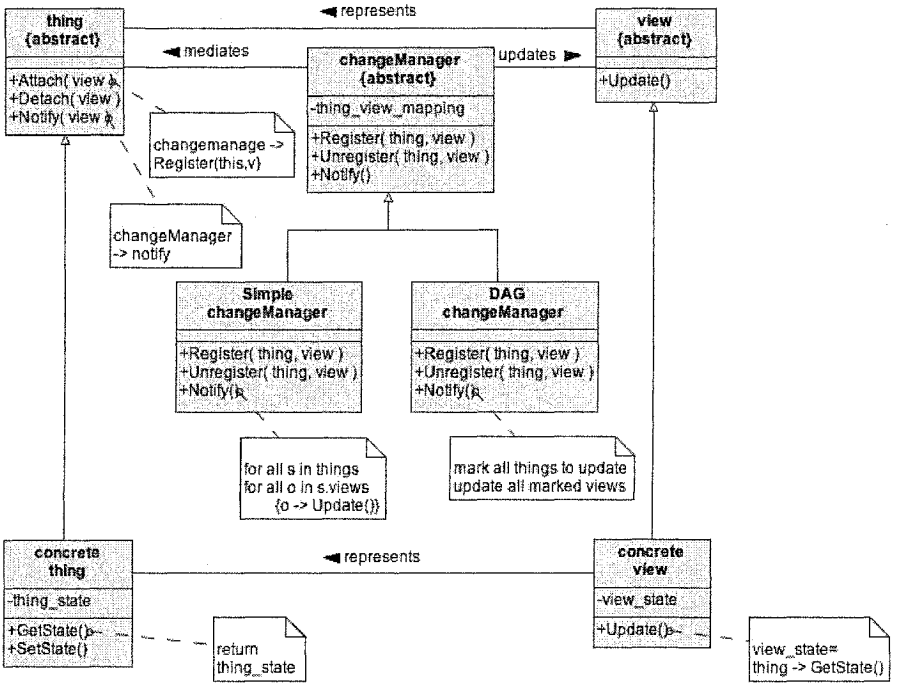

Figure 3 – Implementation mode for views (*interested parties* pattern)

However, while an EO can in principle describe any physical or informational entity, in practice (prEN/DIS 19440) restricts its usage to those entities that are inputs or outputs of transforming processes such as Enterprise Activities. The current template for an EO does not allow the EO to represent some other construct – for example, to be a description of a Business Process. And OVs are essentially low-level mechanisms providing a transient subset of information for selected EOs – they are not intended for and do not provide a general mechanism for obtaining a view on an integrated model[18]. OVs are geared to the limited objective of linking Resources to Enterprise Activities, as reflected in the Figure 4, which is generated from the integrated metamodel that underpins (prEN/DIS 19440), and showing all OV-related relationships:

Consider the situation where it is desired to analyse 'value added' in a chain of processes. This would mean extracting financial information from the processes involved – one could aggregate the costs of the involved resources, but that is only part of the story because it does not include the added value that only the process 'knows about'. Since it is not possible in (prEN/DIS 19440) to obtain an Object

[18] The terminology for Enterprise Object and Object View is unfortunate – CEN TC310 WG1 tried to find terms which did not imply a more general usage but was unsuccessful.

View of a Business Process or Enterprise Activity, it is not possible to build such a 'value-added' view.

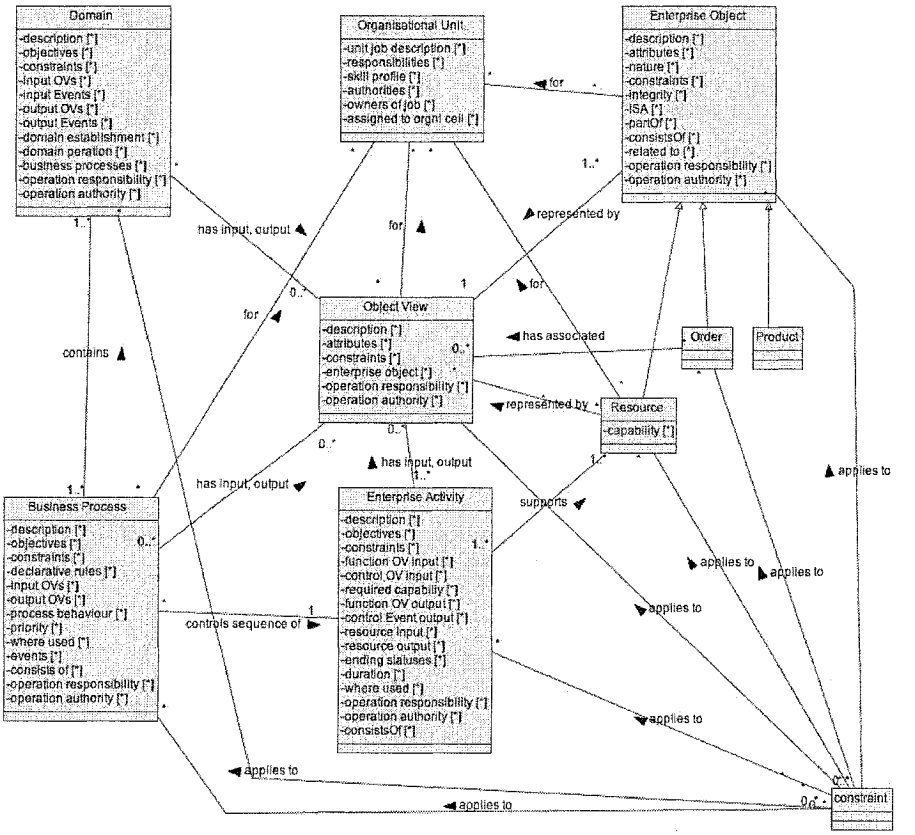

Figure 4 – Object Views and related constructs

Consider also the case where it is required to develop a performance model of processes (this example is suggested by (Salvato 2002)). One possibility is to introduce additional classes for performance and performance indicators, both of which might be associated with Business Process – but it is not clear whether these need to be defined as specialisations of some existing construct, or whether they can be simply attributes of business process. In the former case, they could be treated independently of the process itself, for example, as performance measures that can be associated with several different, but similar, types of business processes.

There is, therefore, a need for a more general 'modelled object' view mechanism, to allow selected attributes of any modelled object or collection of objects to be represented as a named collection. One way of handling this[19] in any revision of (prEN/DIS 19440) would be to define a 'Construct View' as a named

[19] Not considered further in this paper.

collection of selected attributes from a collection of constructs. Object View would then be a specialisation of Construct View.

4. GROUNDING THE (PREN/DIS 19440) CONSTRUCTS IN THE (PREN/FDIS 19439) FRAMEWORK

Several members of CEN TC310 WG1 have criticized the complexity of the early metamodels of the concepts in (ENV 12204) and the later models of (prEN/DIS 19440). The problem arose largely because of an attempt to ground these models in (prEN/DIS 19439) without making that explicit.

To manage this complexity, and to better align with other initiatives, it is now proposed:

- to present the metamodels of 19439 and 19440 as two separate UML packages, called *Framework* and *Constructs*,
- to introduce a new package provisionally entitled *FCM* to define the relations between the Framework and Constructs packages, and
- to define all three packages as metamodels expressed in the UML meta-language[20].

The term FCM is proposed because the (concepts of the) Framework provides meta-concepts for the (concepts of the) Constructs; however, there are no direct entity <–> meta-entity relationships between the concepts of the two models. For example, the construct Resource is not an instance or specialisation of a concept in the Framework – and in particular, an Object View is not an instance or specialisation of the Framework's Model View.[21]

The initial impact of this proposal will be on the treatment of Enterprise Model Views. The argument runs as follows: Given that different modelling situations will require different views, there is a need to separate out the enterprise model itself from the different views that there can be on that model, and to control changes to both to ensure consistency. This is analogous to the Model-View-Controller paradigm now widely used in user interfaces, and to the Observer pattern referred to earlier.

An early MDA working paper (MDA 2001) provides some guidance on how this might be done. "UML provides an important modelling element that is relevant for separating viewpoints, levels of abstraction, and refinements in the context of MDA – the UML package, a UML construct for grouping model elements. A package can import other packages, making elements from the imported package available for its use. Packages help understand the MDA since the models being interrelated for integration are typically in different packages. Models of a system from two different viewpoints unrelated by refinement (or models of two distinct interfaces of a component, or models of two different systems to be integrated) would be defined in two separate packages.

---

[20] Current thinking is that if the packages are legal UML v1.X or later v2.0, then it is not necessary (and would provide little benefit) to define these packages as stereotyped extensions of MOF base classes.

[21] This clarification is derived from the author's interpretation of [Robertson 2004].

"The interrelationships between the two would be defined in a model correspondence ... defining the integration of those two viewpoints or systems at the current level of abstraction." The latest MDA proposals have replaced *model correspondence* by concepts such as mappings, markings and transformations, but the principle remains the same.

This suggests, therefore, defining (within the FCM) a structure for views that holds visibility attributes for each construct in the different views[22] – this is a construct-visibility map, and corresponds to the mappings used in MDA. There is an issue here about non-construct concepts, for example behavioural rules and associations. How is the visibility for these to be handled? Perhaps they should be Visually Representable (VR) *if and only if* any referenced construct is visible? And an association (or constraint?) is VR *if and only if* all its end-points are VR?

Extending metamodelling to (prEN/FDIS 19439) and introducing a new FCM package is seen as the easiest way to formally resolve the meaning of Framework Views on a Construct-derived model23. The alternative would be to explain this in a separate Annex as in (prEN/DIS 19440) or in a separate document as in (Zelm 2001). However, this would not provide an accessible, extensible and maintainable structure for the description of additional views, including those required by a modeller for particular and possible transient purposes. Note that a modeller view may be concerned with something yet to be designed, for which no appropriate view exists beforehand.

The FCM package might also be extended to the (prEN/FDIS 19439) dimension of Life Cycle Phase, so defining and constraining what constructs are visually representable in each phase and what attributes and constraints are visible for modification in each phase. This possibility is not further addressed in this paper, but (prEN/DIS 19440) provides some guidance for this in the A2.x, B2.x sections of the templates for each construct.

5. THE BENEFITS OF FORMALISING STANDARDS AND STANDARDS PROPOSALS THROUGH METAMODELLING

Some informal class models were constructed for (ENV 40003), but solely for developing understanding of concepts during the drafting process. A subsequent analysis by (Petit *et al* 2000) as a contribution to work on drafting prEN/DIS 19440

---

[22] This would need to accommodate both pre-defined and user-defined views.

[23] Extending the existing [prEN/DIS 19440] UML model by introducing new classes derived from the MOF base classes has been proposed by [Salvato 2002]. The specific proposal was to specialise *modelElement* into new abstract classes of *functionElement*, *informationElement* etc, and then to define construct classes as specialisations of those. However that would introduce additional complexity because of the need for multiple inheritance to handle Enterprise Activity and Enterprise Object (both of which are used in more than one View), and it is not clear that it would be sufficient or manageable in supporting extension to new Views (modeller- or even user-defined).

identified several problems and ambiguities in the intended grounding Framework of (ENV 40003) by using the Telos Knowledge Representation Language[24] .

The drafting of the (prEN/DIS 19440) made extensive use of UML modelling techniques, in particular by developing a metamodel of all the constructs in one consolidated metamodel using a UML modelling tool, MagicDraw,™ 25 from No Magic, Inc. One facility provided by the tool (also available in similar tools) was that of defining a new diagram containing a single modelled construct and then adding automatically only those constructs and other modelled elements that were directly associated (through inheritance or association26) with that selected construct. This was a major benefit in checking the consistency of the normative text. Additionally, a concordance program was used to check consistent usage of terms.

While these procedures greatly increased confidence in the rigour of the draft, they are not to be seen as any form of guarantee that the latest draft of (prEN/DIS 19440) is completely error free[27]. While a metamodel can ensure consistency of the things expressed in that metamodel, it does not solve all problems[28] – and manual checking is still necessary against the normative text. Where feasible, metamodelling should be complemented by formal analysis as in the previous Telos analysis of (ENV 40003).

(Holz 2003) argues that basing the formal description of the constructs on the MOF metamodel may (to be proven) assist with the transformation of models based on metamodel formalisms, which are in turn derived from a common metamodel. Previously (Pannetto *et al* 2000) had proposed an approach to generalising two UML fragments, justified by similar semantic description (and presumably a similar usage or context). The IST UEML project found that common concepts can be defined by demonstrable equivalence in a (number of interesting) well-defined context(s) or under some explicit hypotheses (UEML 2003). However, interoperability with other enterprise modelling languages may well require new abstractions to be introduced – for example (UEML 2003) also found that a new class of ExchangeObject was found necessary to represent a common abstraction for process input/outputs in three different modelling languages. An extension mechanism is therefore needed to allow modellers to introduce new generalizations and specialisations, both for classes and (probably) for associations (the latter is a facility to be provided in UML2.0).

---

[24] See http://www.cs.toronto.edu/~jm/2507S/Notes04/Telos.pdf for overview and further references

[25] http://www.magicdraw.com/

[26] This is the reason for representing associations explicitly in additional to the relational attributes captured in the templates.

[27] It is known that in the informative Annex, an association of Event with Enterprise Activity (EA) should show both EA generates Event, and Event initiates EA – however the normative text is correct.

[28] For example, a detailed analysis of the metamodel for UML1.x [Fuentes 2003] identified 450 errors in terms of non-accessibility of elements, empty/duplicate names and derived associations.

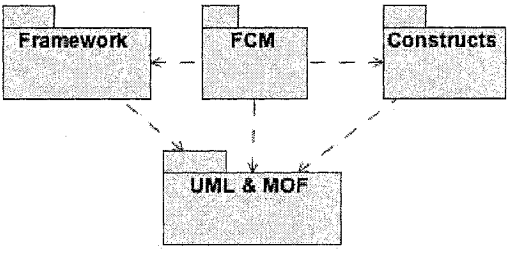

Figure 5 – Packaging the standards

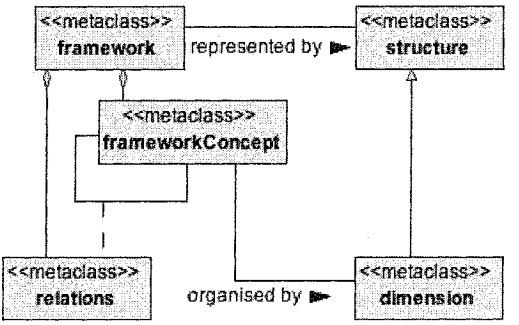

Figure 6 – Organising a framework

6. REVISED CLASS MODELS

6.1 General approach

Figure 5 shows the metamodelling-based approach that is proposed for the alignment of the (prEN/FDIS 19439) Framework and (prEN/DIS 19440) Constructs.

- UML & MOF are essentially givens (meaning that they already exists and will be maintained by the OMG) – however, some care will be needed to reflect any changes made by the OMG as UML 2.0 develops. The other three packages are at very different stages of development:
- The integrated metamodel for (prEN/DIS 19440) Constructs is well developed as an internal document but will need maintenance to accommodate any changes made to (prEN/DIS 19440) after the on-going enquiry and to delete redundant relationships or attributes that were introduced in order to check template consistency. Stereotyping the constructs, associations etc. in terms of MOF base classes would also need additional work if that were seen as necessary.
- Some internal class models were constructed during the drafting of (prEN/FDIS 19439) but these need consolidation and further work.
- The class diagram for the FCM package is still work to be done.

6.2 Class model for (prEN/FDIS 19439) Framework

A framework is a collection and representation of concepts and the relations between them. At its most simple it could be represented as in Figure 6 as a collection of concepts and relationships between those.

View, life cycle, and generalisation/ specialisation are structuring principles which in (ENV 40003) and (prEN/FDIS 19439) are called dimensions.

A dimension is a selective focus from some motivational (stake holder) viewpoint on the domain entities and their different states. It is also a classifier, having some property that enables classifications to be distinguished, for example:

- by being more or less abstract or concrete
- by being more or less general or specific
- by simple enumeration
- by occurring in different phases of something's life cycle.

(prEN/FDIS 19439) defines three dimensions and distinguishing properties for each, as in Figure 7[29].

Figure 7 also proposes *modellingStance* (not a concept in (prEN/FDIS 19439)) for the tuple that represents each possible dimensional combination.

6.3 Class model for (prEN/DIS 19440) Constructs

Annex C of (prEN/FDIS 19440) uses a class model as shown in Figure 8 to describe how constructs are represented in the normative part of the standard using a common template. (Note that the details of the Descriptives and Relationships containers are in general different for each construct.)

A high-level representation of the thirteen constructs is shown in Figure 9 (suppressing attributes and relationships between them).

The Annex then models the constructs in more detail from the viewpoint of each of the model views, each derived from a single integrated metamodel. Figure 10 shows the Function View of the constructs

6.4 A provisional model for mapping constructs into the framework

The approach proposed and illustrated in Figure 11 is to develop an FCM package that conceptually would act as a ViewManager, and as a container for defining what constructs would be used in the existing and new Model Views, further qualified by Enterprise Model Phase and Genericity.

Note that this is not intended as a basis for an implementation (although any such implementation could make use of the *Observer* and *Interested Party* patterns described earlier). Further work is needed in CEN TC310 WG1 and elsewhere to confirm the validity or otherwise of this approach, and to design the details of the FCM package. In particular, the issues of how to handle additional and user-defined modelViews, and detailed constraints and visibility rules for specific attributes are yet to be considered.

---

[29] The issue of how to handle additional viewNames, e.g. decision, economic, user-defined is not addressed in this Figure.

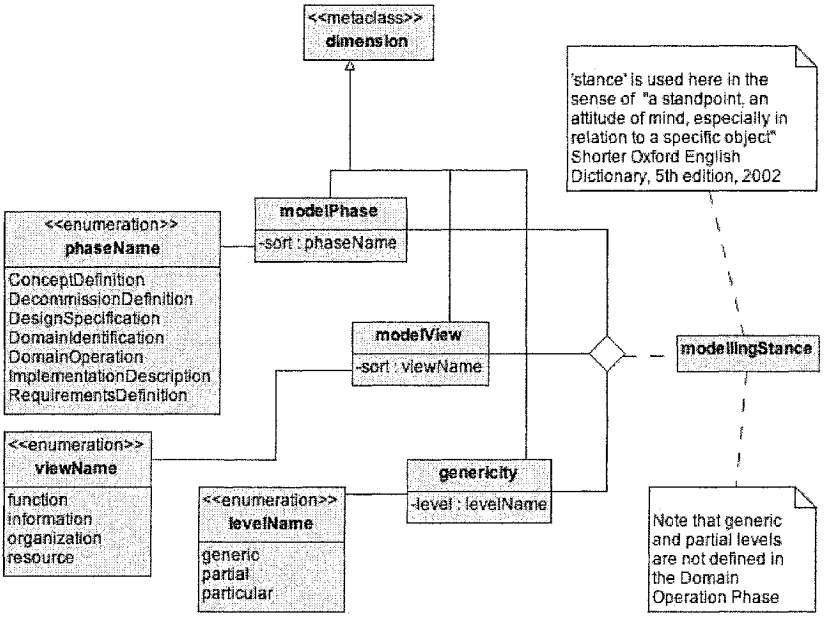

Figure 7 – The dimensions of the (prEN/FDIS 19439) framework

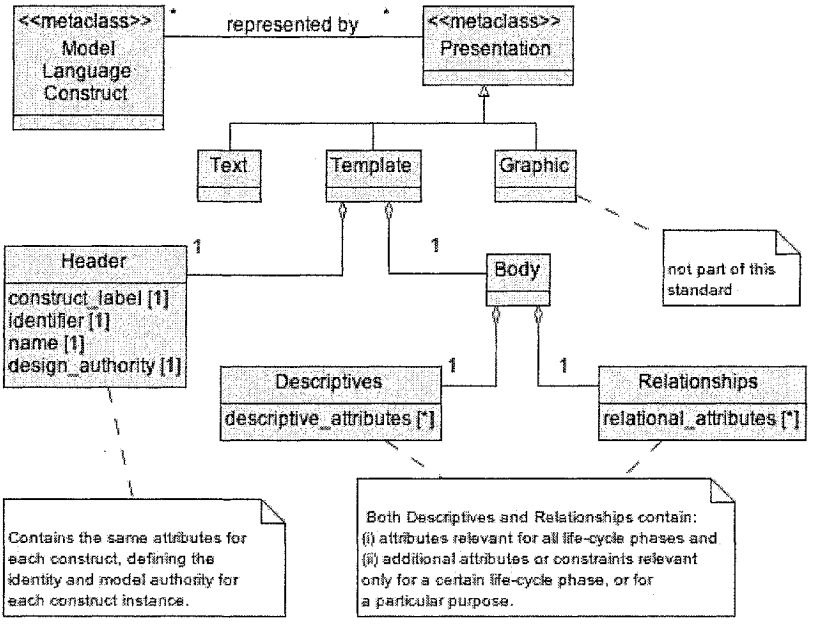

Figure 8 – Template for constructs

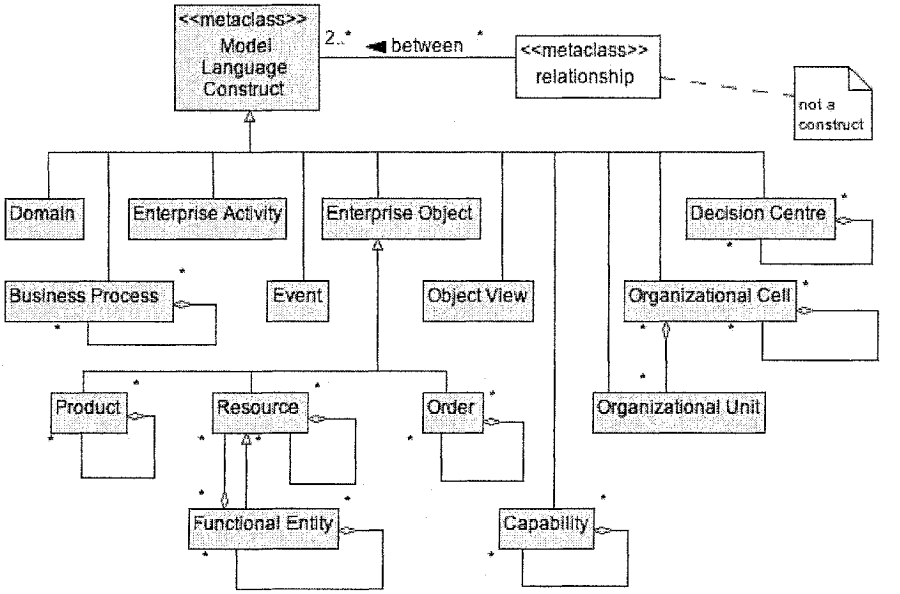

Figure 9 – Overview of constructs

7. CONCLUSIONS

This paper proposes an approach for maintaining the (separately developed and/or modified) *views on an integrated enterprise model* through a change propagation mechanism, based on more formal metamodels of the constructs and framework for enterprise modelling. It sets out the need for a generalization of the current Object View mechanism in (ENV12204) and (prEN/DIS 19440), and suggests how this might be achieved.

Lastly, it argues that using metamodelling to ground the (prEN/DIS 19440) Constructs in the (prEN/FDIS 19439) Framework would provide a route by which standards can be developed to manage current and additional views on an enterprise model, within a federated model development context.

REFERENCES

ENV 12204 (1996) Constructs for Enterprise Modelling, CEN ENV 12204:1996

ENV 40003 (1991) CIM systems architecture framework for modelling, CEN ENV 40003:1991

Flater, D (2002) Impact of Model-Driven Standards. NIST, in Proceedings of the 35th Hawaii International Conference on System Sciences, January 2002, or http://www.omg.org/mda/mda_files/mda_1.7_cleanformat.pdf

Fuentes,J.M, Quintana, V., Llorens,J., Génova,G., Prieto-Díaz,R. (2003) Errors in the UML Metamodel. *Software Engineering Notes*, 28(6). ACM Digital Library 1-13

E Gamma, R Helm, R Johnson, J Vlissides (1997) Design Patterns CD, Elements of reusable Object Oriented Software. Addison Wesley, 1997, ISBN 0-201-63498-8

Holz (2003) E Holz, a metamodel approach for the combination of Models in multiple languages, Modelling, Simulation, and Optimization. 308-313

IS 15704 (2000) Requirements for enterprise-reference architectures and methodologies, ISO IS 15704:2000

ISO 10303 (1994) STEP, Standards for the exchange of Product Model Data, 1994

ISO 14258 (1998) Industrial Automation Systems – Concepts and rules for enterprise models, ISO 14258:1998

MDA (2001) Model Driven Architecture (MDA) Document number ormsc/2001-07-01, Architecture Board ORMSC1 July 9, 2001

MOF (2002) OMG-Meta Object Facility, v1.4, April 2002, © OMG, available via http://www.omg.org/technology/documents/formal/mof.htm

OMG MDA Guide (2003) MDA Guide Version 1.0.1, © 2003 OMG, Document Number: omg/2003-06-01, 12th June 2003

Paige R., Ostroff J (2002) The Single Model Principle, R Paige and J Ostroff, *Journal of Object Technology*, 1(5) 63-81

H Pannetto, F Mayer, P Lhoste (2000) Unified Modelling Language for metamodelling: towards constructs definitions, Proc AIS ,2000 conference, Bordeaux, France, ISBN 960-530-050-8

Petit,M., Ferier,L., Heymans,P. (2000), Some hints for clarification of CEN ENV 12204, Contribution to CEN workshop on Evolution in Enterprise Engineering and Integration, 2000

prEN/FDIS 19439 (2004) Enterprise Integration – Framework for Enterprise Modelling, issued for parallel EN/IS ballot on 2004-09-16

prEN/DIS 19440 (2004) Enterprise Integration – Constructs for Enterprise Modelling, issued for parallel ENQ/DIS ballot on 2004-06-10

Robertson, E (2004) E Robertson, Explicitly Modelling Metadata, private communication

Salvato, G (2002) ENV 12204 metamodel, Contribution to CEN TC310 WG1, (ENV-UML Paper vs2.doc) approx. February 2002

UEML (2003) Common and Non Common concepts referring to the Scenario and GRAI, IEM, EEML metamodels, UEML working paper, IST-2001034229, February 2003

Whitman,L., Huff,B., Presley,A. (1998) Issues encountered between model Views. Flexible Manufacturing and Integrated Manufacturing Conf. Proc. 1998, Begell House, Inc.

Zelm,M. (2001) Representation of Modelling Constructs. Contribution to CEN TC310 WG1, 12/5/01

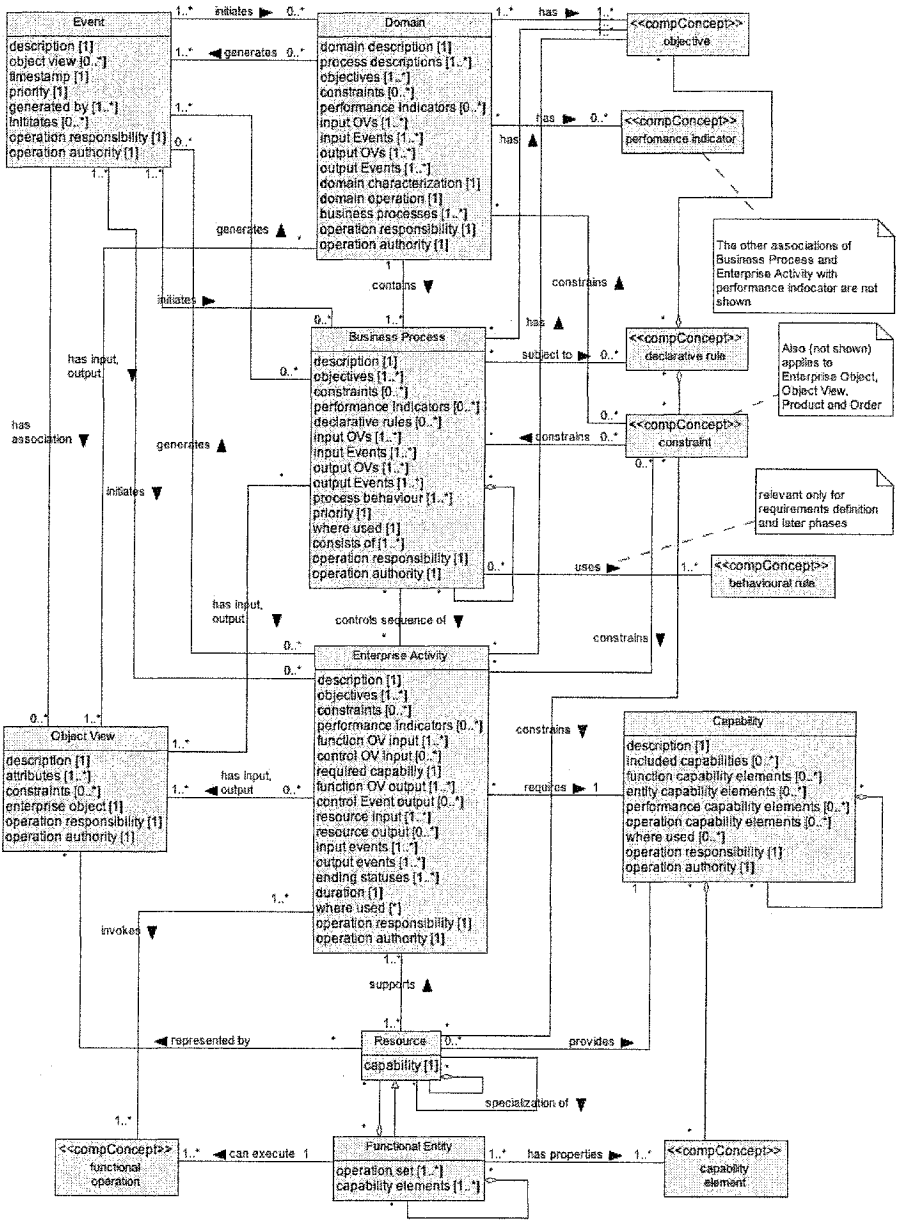

Figure 10 – Use of Constructs in the Function View

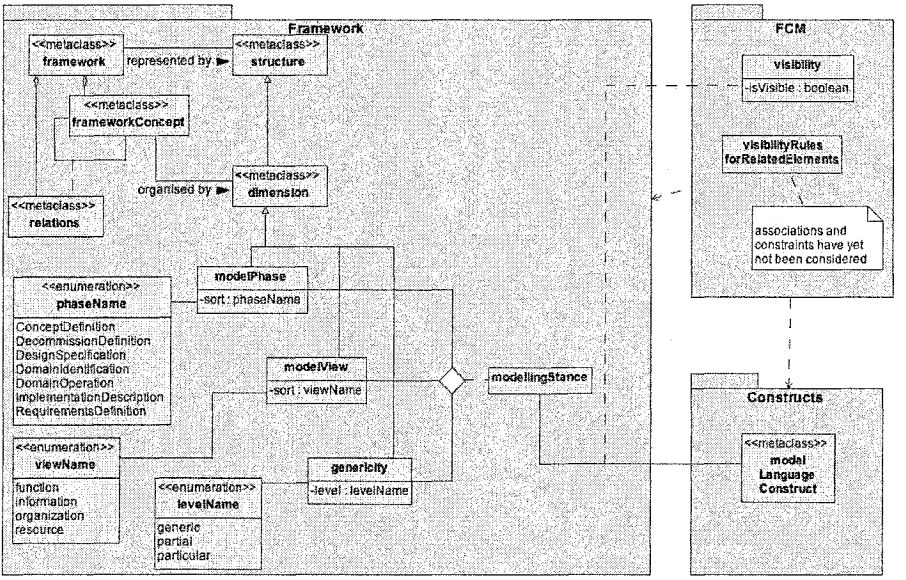

Figure 11 – Role of FCM package