"Meta" Matters

Richard Martin
Tinwisle Corp
richardm@tinwisle.com

Edward Robertson
Indiana University
edrbtsn@indiana.edu

Abstract. Misunderstandings related to "meta" often cause serious problems in systems engineering. This paper attempts to elucidate some of those misunderstandings, particularly as they apply to systems architectures. It examines how standards can provide discipline and clarity, suggesting how standards themselves must be done without falling into similar "meta" traps.

Introduction

A significant barrier to understanding the place of architecture in systems engineering is a massive confusion concerning levels of abstraction with respect to the systems and enterprises we construct. Our focus is on issues of "meta" – the word itself and how we think about and talk about the various levels of abstraction – in systems engineering. While "meta-ness" is but one of many issues in systems engineering and architecture, it is among the least understood and hence mostly likely to cause difficulties.

The impact of confusion about and with meta-levels is especially strong in systems architecture and information systems because both deal with things of the mind and "meta" is entirely a mental construction. System and enterprise modeling is abstraction and "meta" characterizes kinds of abstraction. Hence understanding "meta-ness" clarifies both models and modeling, avoiding the meta-skew which happens when abstraction levels are confounded.

Much of the discussion herein concerns words (and symbols) and how we use and confuse them. This is natural because the problems we address arise because of the way we conceptualize and the way we conceptualize is driven by how we use words.

This paper first examines the many conventional (and not-so-conventional) uses of the term "meta" and how meta-levels form a hierarchy, often confused with other hierarchies. It then mentions some difficulties that arise when this term is misunderstood or misused. Next, the notions of meta-data, a topic of increasing interest, provide a good example of "meta" concerns. Particular connections and mis-connections between meta-ness and several other factors are then considered; these factors include models, architectures and levels of abstraction. The interactions between "meta" and views and "meta" and time are the foci of the next two sections. This paper concludes by suggesting solutions to "meta" confusions and, in particular how International Standards related to system and enterprise architecture can contribute; yet harmonization of these standards depends upon a clear understanding of abstract models and "meta" representations.

The "Meta-" Term

The dictionary gives several definitions of the prefix "meta-"; the relevant one being "more comprehensive, transcending – used with the name of a discipline to designate a new but related discipline designed to deal critically with the original one (meta-mathematics)."(Webster, 2003) A more syntactic definition is "A prefix meaning one level of description higher".(Free On-Line, 2008) In the original Greek, "meta-" meant "behind" or "after", a meaning still found in the anatomical terms such as "metacarpal". The current use arose because Metaphysics came after Physics in Aristotle's work.[1]

"Meta-" is relative; it is an order relationship rather than a property. This is validated by the occurrence of phrases such as "meta-meta-data".[2] When used with a single term, "meta-" is reflective, (but not reflexive, in the sense that "reflexive" is a possible property of an order) with a sense of "aboutness"; "meta-X" means "X about X". Meta-data is data about data, meta-language is language about language,

1. Thus, in being abstracted from physical to cognitive placement, the term "meta-" has itself undergone a meta-transformation.

2. Because "meta-" is directional, it would be convenient to have a term that inverts this direction. Since "meta-" is Greek for "after", we might consider the Greek for "before", which is "pro-" (akin to the Latin "pre"). Unfortunately "pro" already carries way too many connotations.

meta-media is journalists writing about journalists,(Barringer, 2000) and meta-models are models about models.[3] Due largely to Hofstadter's popular *Gödel, Esher, Bach: the Eternal Golden Braid* (Hofstadter, 1979), the reflexivity of "meta" has entered the vernacular meaning explicit self-reference and thus is associated with paradoxes from Epimenidies to Russell and the paradox-like proofs of Gödel and Turing.

"Meta-" often has a sense of abstraction. This is related to the fact that reflection often involves abstraction. For example, a meta-model is an abstraction of a family of models. It is that sense of abstraction which is most important in the following discussion, sometimes discussing "meta" alone and sometimes in contrast to other senses of the word. This two-pronged approach is elaborated in the section on meta-mistakes below.

A distinct, vernacular use of "meta-", which certainly harks back to the notion that metaphysics is considered obscure and incomprehensible, applies "meta-" to anything that is complicated and difficult to understand. Given the importance of the other meanings of "meta" in systems engineering, we certainly hope that it avoids this last meaning.

All Hierarchies are Not Meta-Hierarchies

There are three distinct scales which are important in architecture and architecting (the left of each scale is conventionally considered to be the top):

Abstractness:	abstract	$\leftrightarrow$	concrete
Generality:	general	$\leftrightarrow$	particular
Granularity:	coarse	$\leftrightarrow$	fine grained/detailed

As we noted above, Abstractness is a valid expressions of meta-ness while the other scales are not. These scales are often confused, causing troubling entanglements; hence it is valuable to examine and carefully distinguish the three scales.

First, consider the words. These three scales are named by one of their extremes; that this name is the top extreme for the first two and is the bottom for the last indicates that we most commonly "look up" Abstractness and Generality while we "look down" Granularity. The label "Detail" is often used in place of "Granularity", but adding detail does not always change Granularity. For example, we may add detail to the description of an automobile engine in (at least) two ways: supplying missing facts, such as specifying an engine's displacement, or exposing sub-components, such as block, head, pistons, *etc.* Adding facts does not increase Granularity, exposing sub-components obviously does.[4] The various idiomatic phrases for an action through which granularity is increased – "drill down", "blow up", "expand" – recognize that greater detail is implicitly present but hidden in a "black box".

These three scales are independent. Processes may be decomposed for increased granularity in a Data Flow Diagram (DFD) and entities may be generalized in an Entity-Relationship (ER) model, but these steps do not change the level of abstraction of the DFD or ER model. Nonetheless, it is common to have co-occurrence at the extremes of the scales (for example, a module is concrete, particular, and fine grained).

Contrasting the order structure of the three scales, Granularity has arbitrary but often many levels with tree-like structure (one-to-many along decomposition), Generality has a few levels with a slim many-many structure (building-type and building-style are both generalizations in the realm of civil architecture), while Abstractness has fixed levels with a many-to-many structure that is quite narrow at its upper levels.[5] In tree-structured hierarchies, Granularity and Generality seem to have an obvious inverse relationship, but these must be carefully considered. For example, "all zip codes in HR must be nine digits" is fine grained (highly detailed) in the information model but the implementation of this decision applies generally across the organization.

There are several important questions which should be investigated with respect to these there scales: How does time differ across these? Is each of these useful as a purposive dimension(Martin et al., 2005) in some context? How do decisions differ across these? How do Verification & Validation differ across these? But the focus of this paper is meta-ness, so we leave those issues for examination elsewhere.

---

3. This even fits with the current use of "metaphysics" (but not with the origin of the term) if we recall that physics was once also called natural philosophy, and thus metaphysics is philosophy about philosophy.
4. However, "highly detailed" is synonymous with "fine grained".
5. This seems to distinguish systems architects, who use a restricted set of carefully crafted meta-abstractions, from philosophers, who abstract at will.

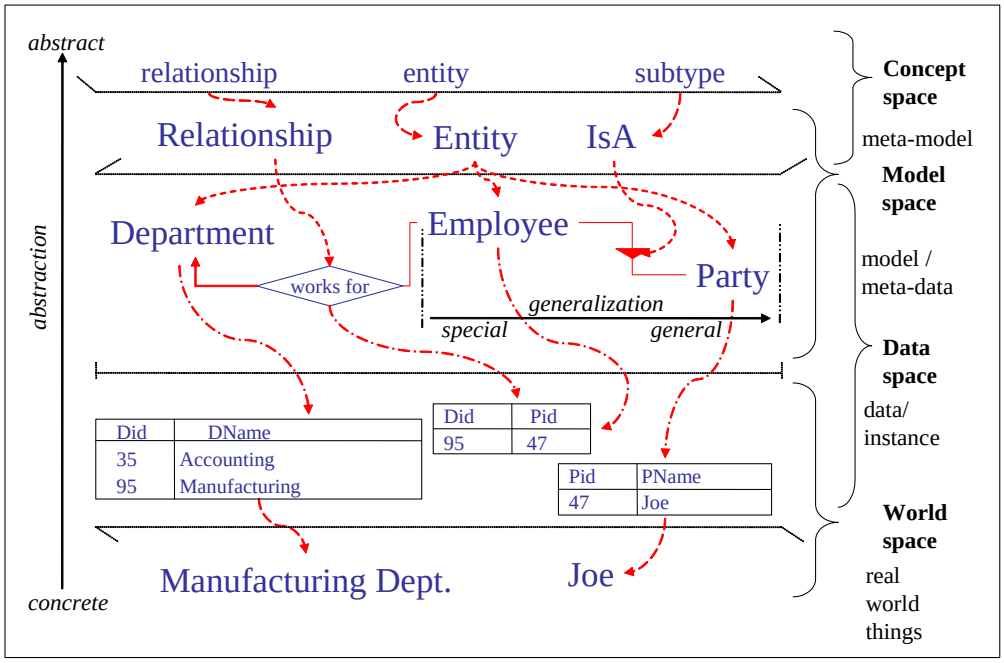

Figure 1. Meta Relationships Example

Figure 1 illustrates the common meta-levels that correspond to abstraction planes. The subject of the example come from an information system, where the bottom is a snippet of the real world captured in the information system, the next layer up is the database proper, above that is an ER model expressing the information model for the database, above that is a conceptual model for ER models, and on top the most abstract concepts through which we build and analyze mental structures. Note that the database level (second from the bottom) is focused on data value instances. The schema for the data base is at the same abstraction level as the ER model; thus indicating a life cycle dimension orthogonal to the page. The interesting characteristic that ER models can express their own meta-model is examined below in Fig. 3 and the associated text. Near the middle of the figure, the example also shows that Party generalizes Employee (as well as other entities not specifically indicated). This example is used because it shows a single theme that is traceable through several meta-levels.

The figure illustrates that a generalization is not an abstraction, even though the thought processes for generalization and abstraction have much in common. The figure also illustrates (right margin) that we typically work in a space that spans two meta-levels, with instances and their abstractions. Modelers and architects commonly work across three levels, since their modeling toolkits are defined one level up and they must model instances one level down. Four levels suffice; that is applying further "meta-"s to meta-meta-meta-reality collapses.

The bottom boundary mapping of the Abstractness hierarchy (the mapping crossing the 'World Space' boundary in Fig. 1, exemplified by the connection between the sign "Joe" and the real-world person identified by that sign) is largely beyond the purview of this article. It is an area that has engaged philosophers for millennia. We only assume, however naively, that the bottom mapping exists; that is, the real world is recognized and recorded.

Meta-mistakes and "Meta-" Mistakes

As the title of this section indicates, there are two kinds of mistakes involving "Meta-". The first is misunderstanding the sense of the prefix or the application of this sense. The second, using the sense of "meta" meaning abstraction, is to confound applications of meta-steps.[6]

6. Note that throughout we attempt to separate the word from its applications, always quoting the word.

An example of the first kind is mere misuse of the term.[7] For example, a `<META>` tag in an HTML document is not a tag about tags but a tag about the document in which it resides – "`<META>`" is really an abbreviation for "`<METADATA>`". The variety of meanings in other contexts adds further confusions.

An example of the second kind occurs in land-use planning. Meta-level confusions can be seen in stark contrast in civic processes such as land use planning because they juxtapose stakeholders with very different understandings of "meta" issues. That is, land-use planning juxtaposes planning board stakeholders, who understand that developing a plan occurs at a different level than quotidian[8] concerns about one land parcel, with citizen stakeholders, who have no appreciation of the subtleties of meta-levels. Thus a citizen, not recognizing meta-level distinctions, may attend a meeting writing a master plan and complain about a neighbor's hogs.

Meta-data

Meta-data is a realm particularly rich in issues of "meta-ness". Thus we focus on this realm in our attempt to understand these issues, with the caution that this is only an exemplar and not necessarily the most important source of meta-ness issues.

Meta-data, as noted above, is data about data. There are different ways in which this "about-ness" may happen: meta-data may be about the occurrence and use of data, about the concepts in data, or about the structure and representation of data. Thus there are three kinds of meta-data. An example of the first kind is the "Dublin Core", the most widely used meta-data standard, which came from the library community and specifies important characteristics of documents, such as creator and publisher. In Geographic Information Systems, meta-data originally recorded the time and means of gathering the data, but sloppy use now characterizes any ancillary data as meta-data. The third kind is most commonly "relational meta-data", the table and attribute names of a relational database. The middle kind is where "meta" as abstraction and "meta-" as "about-ness" coincide and where meta-data transitions into the data model. These three tiers mirror the ANSI/SPARC hierarchy, as shown in Figure 2, which illustrates the parallels with various meta-data applied to a book.

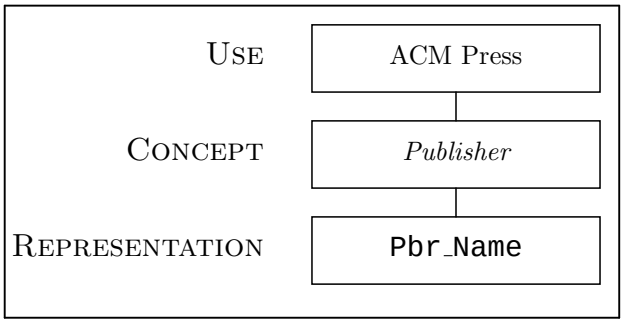

Figure 2. Kinds of Meta-data Parallel the ANSI/SPARC Categories

One of the greatest benefits and gravest dangers of the triple notation used in RDF is its ability to represent meta-relationships in exactly the same way as other relationships.(Robertson, 2004) In so doing, it obscures the above distinctions in the three meanings of "meta-data", as well as any distinctions of meta-level. This requires considerable discipline from semantic web developers so that the meta-levels remain sensibly disentangled. Fortunately, the foundations of the semantic web(W3C, 2002) seem to be constructed using such discipline.

Meta-models, Architectures, & Frameworks

System architectures are of course about the architecting of systems, while frameworks are about architecting the architecture of systems. Thus the architectural framework is meta to the architecture in

7. Note our attempt to be careful in this manner, in as much as a " 'meta-' confusion" is confusion related to meta-ness, while a "meta-confusion" would be confusion about confusion. As is often the case, these are distinct ideas but close enough to cause confusion.

8. "Quotidian" is a single word with a meta-level confusion: the word is unusual, even esoteric, but it means ordinary or commonplace.

the context of a realized system. An architecture is constructed from models and the meta-models are the major components of the corresponding framework.

A framework provides structure. In the real world, this structure is typically also infrastructure, as in a rack or backplane. At higher meta-levels, the infrastructure is conceptual and thus induces another meta-level; an architectural framework provides containment for artifacts and, implicitly, meta-relationships.

Good architecture localizes uncertainty (but does not necessarily limit the consequences) by articulating the relationships between form and function.

The architecture/design distinction is relative, like up/down, not like top/bottom. This suggests that architecture is therefore "meta-" to design: architecture is certainly over design in scope, and indeed we commonly use terminology such as "Gothic architecture" to characterize certain design patterns and their manifestations.[9] An enterprise architecture is thus the (meta-)design for organizing a design process and capturing design artifacts.

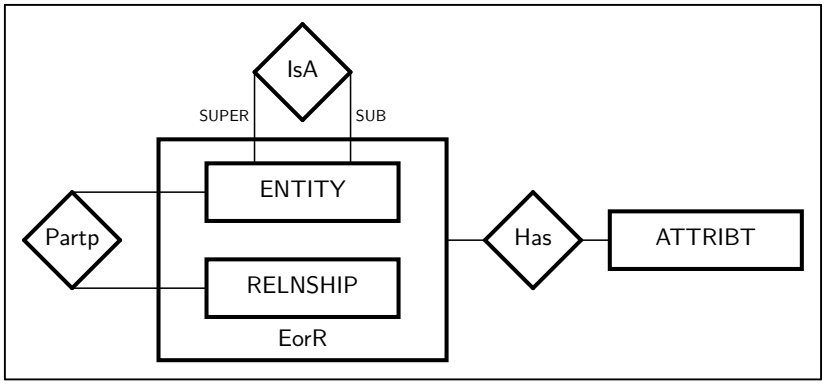

Figure 3. ER Meta-Model for ER Models

The Entity-Relationship (ER) meta-model is itself expressible as an ER model with only three entities, titled `ENTITY`, `RELNSHIP`, and `ATTRIBT` (Figure 3). One of the strengths of the ER approach is that it can provide its own meta-model. This Figure illustrates the reification induced by a meta-level transition, wherein model-level Relationships map to an Entity (that this Entity is named "`RELNSHIP`" illustrates the terminological pitfalls that line the "meta-" path). Note that the ER notion of Relationship reifies to the Entity `RELNSHIP` while `IsA` does not reify to an Entity because `IsA` is not a first-class construct in an ER model. The reification of Relationships may retain the same structural characteristic, which is meta to both a concept and its reification. For example, in Model Driven Architecture, Model Object Facility objects are meta with respect to both Platform Independent Model (PIM) and Platform Specific Model (PSM) levels while the PIM level can be meta with respect to the PSM level.

"Meta-" Confusions

Meta-ness is a very powerful notion, so a misunderstood "meta-" can cause powerful disruptions. This is recognized in a light-hearted manner, as "much hacker humor turns on deliberate confusion between meta-levels,"(Free On-Line, 2008) but where there are jokes there are also serious problems.

"The most common mistake in modeling systems is mixing elements of different levels in the same model,"(Bahill, 2008) for example writing a use case at a high level and a creating a class diagram at a low level. A model which mixes meta-levels, where meta-model characteristics are added to a model, is a major source of confusion because this also mixes internals and externals.

The confusion between "specialize" and "instantiate" is often a major hurdle in modeling. This confusion is cleared up by examining the meta-levels involved: specialize is same-level of abstraction while instantiate is down-level to more concrete. Applying these notions to an example involving the PERSON entity:

PERSON $\rightsquigarrow$	EMPLOYEE	(specialization)
PERSON $\rightsquigarrow$	John Smith	(instantiation)

9. While individual styles, such as "Gothic" or "Art Deco", are generalizations, the concept *Style*, in itself, is an abstraction.

This confusion is related to the ambiguity of "is a": the above examples may be read right-to-left as "John Smith is a person" and "an employee is a person".

Transitions to meta-levels do not necessarily go in parallel. For example, an information model may model both data and meta-data, but the presence of that meta-data does not make such a model a meta-model. The above-mentioned parallel between the architecture → framework transition and model → meta-model transition is a rare exception where two meta-steps coincide.

The most problematic confusions associated with meta-transitions arise because any specify/design/implement methodology transitions up from the problem space and later back down into the solution space. Problems occur when abstractions (targets of meta-transitions) in the solution space get confused with abstractions in problem space; such a confusion explains the push to characterize all problems in terms of objects (solution space abstractions). Even worse, people often confuse understanding the solution with understanding the problem. Because 'the solution' often involves one local fix in one given context, the result is poorly founded and brittle. Yet people get paid for providing a solution.

The above discussion broaches the intertwining of meta-steps in processes. Throughout a system's life cycle, there is propagation from meta to instance and feedback from instance to meta. It is along these "roundtrips" that confusions arise – their consequences may not be felt for many iterations.

Finally, constraints at the meta-level ideally should not constrain the solution. This ideal is often violated, as when the choice of a programming language or development environment makes certain solutions harder or easier (people did write compilers in COBOL, but doing a modern code optimizer that way is terrifying to contemplate). It may be necessary to say "we must implement in C because ...", but the impacts must also be acknowledged at the meta-level. This is essentially the old commandment "design with your tool in mind."

Views & Meta-ness

The notion of View is very general and is "meta-" (in the abstraction sense) to any model, standard, or framework. Most problems with views and meta-ness in standards occur when a view consideration incorrectly span meta-levels.

Views are and the viewpoints that define them are essential features of standards, where there is a growing use of views to capture the significance of stakeholders' concerns as applied to an architecture.[10] Standards commonly have a collection of mandated views that derive from the concerns of certain known classes of stakeholders. Such views commonly appear corresponding to a projection along some dimension of the underlying framework. For example, an *information view*, which is a projection along the analog of the `WHAT` interrogative in a Zachman framework, is commonly mandated. The phrase "corresponding to" above is deliberate, since standards typically do not have an explicit view mechanism.

However, calling one dimension "View" ties this top-level "meta-" characteristic into a lower level. For example, the "Genericity" dimension of ISO 15704(International Organization for Standardization, 2000) reflects a meta-model characteristic, in that a Generic, Partial, or Particular view (that is, a slice through the model space along one coordinate of the "Genericity" dimension) is a complete and coherent framework (in the Zachman sense) with a degree of specificity appropriate to that coordinate. A slice along a coordinate of another dimension, say the Information View within the Model View dimension, does not produce a framework but merely a collection of artifacts with comparable content but differing in specificity. Therefore, a Model View view exhibits internals of a framework or set of frameworks. GERAM(IFIP-IFAC Task Force on Architectures for Enterprise Integration, 1999) calls this particular dimension "View", which is even more confusing. Because the components of each point on GERAM's "View" dimension are in fact entire architectures, GERAM is in fact an architecture framework rather than an architecture. ISO 19440,(International Organization for Standardization, 2007) which presents a Generic model, reflects a solution to the above problem.

A view expressed at a particular meta-level projects down to lower meta-levels. Indeed, this is so common a mechanism for expressing views that we do it without being aware that we are doing it. For example, an SQL `VIEW` is defined on tables but applies to instances. On the other hand, a view projected up a meta-level is very rarely informative or even non-trivial. The upward view from any ER model yields the same meta-ER model, as noted above.

The meta-levels of views are typically specified as if they are distinct, while the instance levels of these views commonly overlap. For example, in a Zachman frame, a model (meta-level) in the `WHAT`

10. The term "viewpoint" is often ambiguously used both to encapsulate a set of concerns and to specify how to create a view addressing these concerns.

column will indicate the abstraction of an artifact and the HOW column will indicate a process; but the actual (instance-level) process applies to the actual artifact.

One would hope that a view through a Model Phase,(International Organization for Standardization, 2006) say Operation, would be coherent. But it may not be complete historically.

When a view crosses meta-levels, view updates become very problematic unless very carefully controlled. In the database world, updating table structure while updating data could cause chaos. In common practice, allowing only database administrators to perform meta-updates provides the necessary control.

Meta and Time

Time interacts with "meta" because processes are accomplished over time. But there are different time spectra at different meta-levels. And these spectra may have very different properties. Returning to the land use planning example introduced above, the processes of urban growth and rural property development have a very long time frame (in US, since at least the initial times of European settlement), while related meta-processes, namely the development of new land-use plans and related ordinances, happen recurrently over comparatively shorter time periods. Hence the base processes (growth and development) are approximately continuous while the meta-processes have major and specific step functions, namely when planning begins and when new ordinances come into effect.

Constraints are easy in a static system, while time induces changes requiring (re-)evaluation of constraints. We tend to think of architectures, once completed, as static. But, at the meta-level, architecture use changes over the course of a project. This is a consequence of the change in stakeholder emphasis, made more serious because the variety of stakeholders populating various meta-levels often goes unrecognized. That is, an architect's stake in a project is at a higher meta-level than that of a worker who operates the resulting system.

Time relates the Abstractness scale to Granularity. Obviously, models increase in refinement and detail over time. But the constraints and decisions used to verify and validate those models also become increasingly fine-grained over time. Without recognition that these two are on separate meta-levels, validation of the system development process becomes entangled with operation. On the other hand, business process establishment and business process improvement are on the same meta level but at different time points.

Understanding the different time spectra will avoid such knots, for example the constraint in a petrochemical factory, that a particular reactor vessel "must always be at least half full" without allowing for the fact that the vessel is necessarily empty at start-up.[11]

Towards a Solution

The problems with meta-ness are human problems. These problems occur in the way we conceptualize and describe. Thus any solutions must address the human thought patterns. The first step is to spread awareness and understanding of the issues relating to meta-confusions. We hope that the analysis we have done above contributes directly to this awareness and understanding. The single most important factor is simply the warning: *Watch out for 'meta-' issues.* This appears deficient in specificity, but that is in fact its greatest value. As we have seen, the meta-monster is a chimera, able to assume many different forms. Warning "watch out for snakes" seems insufficient when someone is subsequently stung by a scorpion.

Perhaps the hardest thing in systems analysis is extracting what the users need from what the users say. Often this is because users freely wander across the "meta-" boundary. Thus analysts must take special care with "meta-" issues.[12] This is even more true for standard authors, since they are meta-analysts.

A more powerful tool is found in standards. Standards can and should enunciate general warnings, but having standards writers heed these warnings is much more effective. Several general notions must be manifested is the standards, including:

---

11. This, of course, is related to the fact that the implicit universal quantifier in that statement creates a time variable which is in some sense meta over the time instances. Thus meta concerns return to meta-mathematics.
12. However, care with "meta-" cannot resolve non-meta issues, as when Joe wants a user interface that looks more impressive than Sue's.

- To create coherent models, the "meta-ness" of model content must be consistent for all scales of elaboration.
- The usefulness of models to describe collections of systems is relative to coherence across scales of elaboration.
- Standards target different extents of elaboration, with architecture standards focusing on the modeling domain as much as the application domains.
- The higher the meta-level of a standard, the more attention that standard should pay concerning meta-ness issues.

Lack of Generic or Partial models in the Operation Phase of ISO 19439(International Organization for Standardization, 2006) is a consequence of the fact that during this phase a meta^{-1} transition is essential, where every such transition is grounded in particulars. A model, once in operation, is fixed and its "metas" are irrelevant to that operation. In fact, conceptual changes to an executing model for production occur by revisiting the prior phase of the life cycle. Thus the Operation Phase of ISO 19439 contains no Generic or Partial models. But when standards dictate that "metas" must be considered, changes are required to be made at the design level rather than "on-the-fly".

In January 2008, the ISO TC184/SC5/WG1 participated in the 2008 workshop of the International Council on Systems Engineering (INCOSE) with the objective of gathering input for standards revisions. The draft report resulting from this workshop includes statements such as "We must distinguish the life cycle through which the architecting happens from the life cycle of the architected system; standards need to address both."[13] The requested distinction is one of "meta"'s.

Standards all have multiple facets of meta-ness. Naturally standards are "meta-" with respect to their target domain. Some standards, like ISO 10303 and ISO 15745, produce an initial meta-model and then provide a mechanism for further customizing meta-models for specific sub-domains. This mechanism is then "meta-" with respect to the other meta-models. We note an emerging trend in such standards toward an XML expression of the particular models that may reduce the effort required to implement the standard by allowing an XML meta-model engine to directly implement compliance in the product. Finally, standards often specify compliance mechanisms which are "meta-" to other aspects of the model. This complexity suggests the need for a thorough "meta-meta" analysis.

Finally, observe that architecture standards are themselves meta, in that they govern how other standards are described and articulated. An architecture standard might explicitly note that it "will not define X", where X is a model representation, consistency criteria, or other architecture factor. However, it should require that an architecture conforming to that standard do so. The issue is placing requirements at the correct meta-level. In general, an architecture should provide clear decision support as architecture transforms into design, but this has an inherent meta-knot.

References

Bahill, A. T., 2008. untitled web page. accessed March 2008.

Barringer, F., 2000. A death is noted in the meta-media family. *New York Times*. (June 5, 2000).

Free On-Line, 2008. The Free On-line Dictionary of Computing. (accessed Feb 26, 2008).

Hofstadter, D., 1979. *Gödel, Escher, Bach: an Eternal Golden Braid*. Basic Books.

IFIP-IFAC Task Force on Architectures for Enterprise Integration, 1999. GERAM: Generalised Enterprise Reference Architecture and Methodology.

International Organization for Standardization, 2000. Industrial Automation Systems - Requirements for Enterprise–Reference Architecture and Methodologies (ISO 15704:2000). `www.iso.ch`.

——, 2006. Enterprise Integration - Framework for Enterprise Modeling (ISO 19439). `www.iso.ch`.

——, 2007. Enterprise Integration - Constructs for Enterprise Modeling (ISO 19440). `www.iso.ch`.

13. As this paper goes to press, the INCOSE report is still in draft form.

Martin, R. A., Robertson, E. L., and Springer, J. A., 2005. Architectural principles for enterprise frameworks: Guidance for interoperability. In Bernus, P. and Fox, M., editors, *Interoperable Strategies for the Enterprise Architect*, pages 79–92. Springer. `www.cs.indiana.edu/ftp/techreports/TR594.html` has an expanded version.

Robertson, E. L., 2004. Triadic relations: an algebra for the Semantic Web. In Bussler, C., Tannen, V., and Fundulaki, I., editors, *Semantic Web and Databases*, pages 91–108. Springer. `www.cs.indiana.edu/ftp/techreports/TR999.html` has an expanded version.

W3C, 2002. RDF Model Theory. `www.w3.org/TR/rdf-mt/`.

Webster, 2003. Webster's dictionairy, 7^{th} edition. online version.

Authors

Richard A. Martin is president of Tinwisle Corporation. After 15 years of providing technical direction for automated laboratory instrumentation at Indiana University, he has for the last 29 years provided information systems services focused on enterprise integration to companies in the manufacturing, distribution, and services sectors. He is convener of ISO TC184/SC5/WG1 (International Standards Organization Technical Committee 184 - Industrial automation systems and integration, Sub-Committee 5 - Architecture, communications and integration frameworks, Working Group 1 - Modeling and architecture).

Edward L. Robertson recently retired as Professor of Computer Science and Informatics at Indiana University, where he had served as chair of the Computer Science Department and Associate Dean of the School of Informatics. He was founder and Executive Director of the Indiana Center for Database Systems. He has taught software engineering and information systems for over 25 years and has published papers in many facets of those areas, including data modeling, formalisms and formal properties, data mining and visualization. In the realm of "meta", he has, with his students, developed a proper extension of relational algebra to incorporate relational meta-data, characterized approximate functional dependencies in the meta-structure of database tables, developed algorithms to mine for that structure, and developed an algebra which elucidates how RDF can express structure across several meta-levels.