Conceptual Meta Model for Building Information Modeling

Vladimir Nývlt
Institute of Business and Technology in České Budějovice, Okružní 517/10, 370 01 České Budějovice, Czech republic
vlad.nyvlt@gmail.com

Abstract. The first meta-concept reflects a specification for data as a whole. We call this a Data Model. Good examples are ontologies or dictionaries. In addition, we have Data Sets that contain individual data according to the Data Model. The next basic meta-concept inside such a data model is a Concept referring to abstract notions as types of things of interest. Next we have Attributes being able to describe intrinsic characteristics and Relations to describe extrinsic characteristics of concepts. Concepts can be instantiated with Individuals referring to real world things you can or could point at. Such instances get lexical or reference Values for attributes respectively relations. Lexical values can be classified according to some Value Type (like string, decimal, integer, boolean etc.). Concepts can have Constraints restricting the amount of values or the values themselves or both. Also attributes and relations can have restrictions with respect to their source or target concepts (in case of relations) or value type (in case of attributes). Finally we have Derivations for concepts that tell us how new values for attributes or relations can be inferred from existing asserted values.

We define three mechanisms as specific relations:
1. Classification (inverse: instantiation), from ‘concrete’ to ‘abstract’
2. Generalization (inverse: specialization), from ‘specific’ to ‘generic’
3. Composition (inverse: decomposition), from ‘detailed’ to ‘global’

These three mechanisms generate three hierarchy types namely a typology (of concepts), a taxonomy (of concepts, attributes or relations) and a meronomy (of concepts) respectively. Concepts are themselves instances of ‘Concept’ and can be instantiated in instances; Value Types are themselves instances of ‘Value Type’ and can be instantiated in values. Concepts can be specialized in other Concepts; attributes can be specialized in other attributes, and relations can be specialized in other relations. Specialized concepts, attributes and relationships inherit all constraints and derivations of the concepts, attributes, relationships they are specialized from. Concepts can be decomposed in concepts; Instances can be decomposed in other instances.

Beside the three abstraction mechanisms we also provide other relation types:
• Grouping, like for grouping all concepts, attributes, relations etc. into one data model,
• Characterisation, indicating how attributes, relations, constraints and derivations relate to concepts/individuals on concept/instance level, and
• The general associations on both concept and individual level.

1. Introduction
The main feature of this modelling issue is the choice for the W3C Linked Data/Semantic Web (LD/SW) [LD-SW] approach and technologies, providing us with the basic formats, access methods and languages. This can be considered as the “conceptual modelling supply side” for fulfilling the demand side from the Asset Life cycle Information Management (ALIM).
Most of its advantages stem from the fact that this LD/SW approach is fully Internet/WWW-based since it is defined by the W3C and the Internet/WWW is utilized as the underlying communication infrastructure. In short: W3C took their existing WWW, being itself already on top of the Internet, and added ‘computer-processable’ (Linked Data) and, next, ‘computer-interpretable’ (Semantic Web).

Linked Data is a concept defined by the World Wide Web Consortium (W3C) positioned on top of the existing World Wide Web (WWW), itself based on the Internet. The data on the WWW is often unstructured and configured for human interpretation. Linked Data is structured and therefore ‘machine-processable’ by software applications. Another layer of semantics can be defined on top, with ontologies containing concepts, datatypes, properties, relations and constraints & rules, giving powerful meaning to the data making it “machine-interpretable”.

This four-layered ‘internet protocol stack’ is shown in the next figure, demonstrating the evolution of the internet as a communication infrastructure from linked computers, to linked documents, towards Linked Data (LD) and then applying Semantic Web (SW) technology to enhance the LD with knowledge of the things of interest relevant in the end-user’s domain [1].

![Figure 1. LD/SW on top of the Internet/WWW.](image)

2. **Four perspectives on BIM data to be modelled**

1) Why is the data needed?
   Business aspects: Multi-party asset management processes supported or even enabled via the data needed. Typically this involves the whole life-cycle and supply-chain of an asset covering a variety of disciplines and from strategic (goals), tactical (means) and daily operational viewpoints.

2) What kind of data is covered?
   Conceptual aspects: what data models / data sets are modelled and linked.

3) How is the data formatted, modelled, accessed, verified, inferred, and delivered or shared within or between parties?
   Again conceptual aspects but now from a ‘solution’ side. This includes the choice of conceptual meta-model, data modelling languages, modelling styles, identification and naming conventions, etc.

4) Where is the data software-wise stored, accessed and processed?
   Implementation aspects: software platforms and tools used for data storage like Data Base Management Systems (DBMS’s), data access mechanisms like Application Programming
Interfaces (APIs), Query Languages (QLs) and generic software tools like reasoners, verifiers and visualisers.

**Figure 2.** Four perspectives on data

Such standard focuses on the (semantic) modelling and linking methodology of asset/project/product data [2]. Implementing software following this standard will result in powerful and interoperable data models and data where data can be easily exchanged and/or shared between parties involved in the assets life cycle and supply chain. In the project management phase an asset is programmed, design and built using existing products from the supply-chain as much as possible. In the asset management phase it is used, maintained & renovated and finally demolished & recycled.

3. **Ontology and System Engineering**

An ontology is a shared abstract view of a part of some real-world domain used for some specific purpose. An ontology is essentially a set of concepts, instances, value types, values, attributes, relations, constraints and derivations. Typically, a taxonomy or sometimes a meronomy, or both, constitute the ‘backbone’ of an ontology.

In the language of the W3C Linked Data (LD) / Semantic Web (SW) approach, an ontology can be rephrased as a collection of classes, individuals, datatypes, literals, datatype properties, object properties, constraints and rules. An ontology generally does not contain individuals except for reference individuals, modelling, for example, standard enumeration items as allowed values. Typical, generic relations are 'specialization' on the class level and 'decomposition' on the individual level, the latter limited on class level. These two generic relations give rise to a Taxonomy respectively a Meronomy (a ‘typical decomposition’) on class level. The term ‘Object Type Library (OTL)’ is seen as a synonym for the term ‘Ontology’ [2]. More about ontologically funded profile for conceptual modeling in Guizzardi [3].

System Engineering (SE) on the other hand handles data of assets going through their life cycle. It describes an asset from the views from different life-cycle phases and the interrelations between those views. A distinction is often made between a functional description (related to functional requirements and technical boundary conditions) and a technical description related to 'solutions' that meet those requirements in terms of performance), both of which can change over the time. In both cases, both physical and spatial aspects of the asset are important. The supply chain dimension is closely related to the decomposition (“system breakdown”) of an asset (like: built environment, networks, assets, elements, components and materials). The SE view can by nature interrelate the more specific BIM and GIS views.
4. Semantic Modelling and Linking
This approach addresses the following topics:

- **Conceptual Meta-Model (CMM)**
  - Format/Language-independent

- **Language-bindings (“Modelling Styles”):** mapping of CMM to language constructs supporting various ‘Levels of Capability’ (LoCs)
  - Providing a choice of RDF data serialisations/formats
  - Subsets of standard Linked Data/Semantic Web languages

- **Conceptual Model (CM), incl. language-bindings via 2.**
  - Top Level
  - Identification, naming, annotation, enumeration datatypes, complex properties (attributes and relations), quantities and units, decomposition, etc.
  - Linking / Link Sets (on data-level & ontology-level)

4.1. Conceptual Meta Model (CMM)
The language-independent Conceptual Meta Model (CMM) from the ‘demand side’ is given in the next figure.

![Conceptual Meta Model (CMM)](image)

**Figure 3.** Conceptual Meta Model (CMM).

The first meta-concept reflects a specification for data as a whole. We call this a Data Model. Good examples are ontologies or dictionaries.
Finally we have Derivations for concepts that tell us how new values for attributes or relations can be inferred from existing asserted values.

We define three mechanisms as specific relations:
- Classification (inverse: instantiation), from ‘concrete’ to ‘abstract’
- Generalization (inverse: specialization), from ‘specific’ to ‘generic’
- Composition (inverse: decomposition), from ‘detailed’ to ‘global’

Beside the three abstraction mechanisms we also provide other relation types:
- Grouping, like for grouping all concepts, attributes, relations etc. into one data model,
- Characterisation, indicating how attributes, relations, constraints and derivations relate to concepts/individuals on concept/instance level, and
- The general associations on both concept and individual level.

Attributes can be further divided in annotations, adding human-interpretable IDs, names, labels, etc. and computer-interpretable qualitative and quantitative aspects [1].

4.2. Top level meta-modelling

A top-level is often heavily influenced by abstract ideas from philosophy, physics, mathematics and logic. Whatever top-level is defined in the end, it is important to remember the following key design principles for it:
- Relatively complete (everything to be modelled in the end has a placeholder)
- Explainable
- Extendable
- Logically coherent (all items independent/orthogonal/no overlap in semantics, important also when applying reasoners consistently later)
- Practicable/handy (recognized concepts in current modelling practice/initiatives)
- Unbiased (e.g. when an ‘activity’ is modelled it is not only meant as being a human activity etc.)

It is with these principles in mind that we have to carefully analyse if certain often tricky concepts like “Role” or “Function” should be added to the top-level taxonomy or not. We have to make sure that we do not harm the principle of ‘logical coherency’ by having the item already implicitly in via other meta-concepts. An example to clarify: sometimes a distinction is made between ‘normal classes’ and ‘role classes’. Normal classes reflect a concept by its ‘nature’ where a role class is heavily influenced (or even: earns its existence) by its relationship/interaction with other concepts [4]. Typical example: A ‘Person’ is a normal class. A ‘Father’ is a role class. An ‘Employee’ is another role class. For now it is decided that such a role class is ‘just’ a subclass of the relevant normal class where there is a constraint on some relation In this case a Father and an Employee would be subclasses of Person with a min-cardinality constraint on the relation ‘hasChild’ resp. ‘worksForCompany’. When things can play multiple roles, or the other way round, roles can be played by multiple things, care have to be taken when specialising since identities might not correspond.

Instances can be typed according to (classified as) to multiple classes, one or more normal classes but also one or more role classes (one can be an employee and a father, hence a person for these two reasons). Finally, a top-level can also apply to attributes or relations (not only classes).

This approach is a variant of the Ontology for Property Management (OPM) [5]. OPM defines three levels. Level L1 uses direct data type properties. Level L2 adds one indirection through objectification of assigned attributes/relationships with a value, and level L3 adds a second objectification for the requested property value. Our variant does objectification only once, but then for the property value (which we call quantity value here). It is therefore a combination of L2 and L3. Furthermore OPM
uses CDT (Custom Datatypes) [6] / UCUM (Unified Code for Units of Measure) [7] for quantities and units.

We predefine a top-level for concepts, attributes and relations (mostly associations). The top-level concept taxonomy has seven same-level sub-concepts under a root (most generic) concept referred to as a TopConcept:

- TopConcept
  1. PhysicalObject
  2. InformationObject
  3. Activity
  4. State
  5. Event
  6. SpatialRegion
  7. TemporalRegion

A distinction is made in objects (subdivided into physical objects and information objects) that ‘ARE’ and activities that ‘HAPPEN’. The meta-model introduced earlier does not yet introduce ‘time aspects’ so this has to be taken care of by the actual modelling itself, here by the introduction of concepts beyond the those static objects: states & events. When time is not relevant because we want to model timeless static aspects or just one ‘snapshot’ of objects in time, states and events can be ignored. This is often referred to as the “3D” approach. Whenever we want to model multiple snapshots in the same model we need both the state and event concept. This is often referred to as the “4D” situation (or 3.5D in case 4D is reserved for a more ‘time-continuous’ situation).

There are two main relationships between objects and activities: physical objects perform and transform activities.

Both objects and activities live in spacetime so both could have a relevant interior and boundary in space and a relevant time period (via spatial region respectively time period).

A physical object covers both materialized objects and functional/mental/design objects. A good example of a functional object is a transport network like a road network. Between such (sub)types of physical objects often a ‘realises’ or ‘fulfils’ relation is modelled (not predefined here). A transitive variant arises where materialized physical objects decompose into sub-functional objects.¹ This idea has in principle the exact counterpart on the activity-side although less applied (involving physical activities fulfilling functional activities like a bridge that performs by functionally ‘connecting’ in time).

A spatial region is used for two purposes: defining the topological interior and the boundary of a physical object. For instance, by specifying the boundary of a functional transport network a 3D corridor network is defined. This is accomplished by aggregating the boundaries of the edges within the network which themselves are topologically connected via nodes.

On the attribute-side we only predefine a root (most generic) attribute: topAttribute.

Next, a top-level for (possibly multi-lingual) annotations is given by:

- labels
- codes
- comments

¹ A idea introduced earlier by Wim Gielingh of TNO under the term ‘Functional Unit / Technical Solution (FUTS) decomposition.'
Finally, the Top Level has ‘same-level’ relations (associations), restricted to their source and target concepts including class-constrained variants of the composition/decomposition mechanism (also their inverses are given).

Figure 4. Top Level relations

When we graphically combine the top-level concepts and associations we obtain graph representation of relations between Digital Twins. Physical object as real entity, real object, and Information object as Digital Twin.
Figure 5. Relations of Digital Twins as predefined concepts, attributes and relations.

Other ontologies can expand on these generic concepts and relations like via further specialization:

- **PhysicalObject**:
  - **Asset**
    - **Building**
      - ResidentialBuilding
      - UtilityBuilding
      - Office
      - School
      - Hospital
      - Factory
    - **InfraObject**
      - RoadPart
      - CivilStructure
        - Bridge
        - Viaduct
        - Tunnel
        - Dike
        - Sluice
        - Embarkment
    - **Environmental objects\(^2\)**
      - TrafficObject
      - Vehicle
      - ClimateWeather

\(^2\) As loads/supports for assets or as functional reason why we have assets anyway.
- Ground
  - many other related things in the right place to be determined like:
  - Degradation
  - Damage
  - Risk
  - Measure

5. Conclusion
Effective management of building and construction assets over their life-cycle requires a smooth flow of high quality information. Standardisation in asset information modelling and management will help to create a functional, safe, sustainable and economic built environment. Construction parties including asset owners/managers, contractors and engineering firms, need to be able to combine international information standards, such as for BIM, GIS and Systems Engineering, with national standards, such as classification systems. This way they can systemize and align their data-related activities and communication.

In essence, an RDF model is a so-called ‘directed labelled graph’ made up of triples of the pattern <subject predicate object>. These triples are the ‘data quanta’ of all RDF models. A triple is never edited, just created or deleted, without any unwanted side-effects. Merging RDF data simply means throwing all triples together and checking for logical consistency. In ‘relational terminology’ one could say that RDF provides maximal conceptual normalization.

Because RDF/RDFS/OWL/SHACL reflect some formal logic system; some fragment of First Order Logic (FOL), data is not only asserted but can also be derived/inferred by generic reasoners that implement this specific FOL fragment

References
[1] Nývlt V and Novotný R 2018 IOP Conference Series: Materials Science and Engineering Vol 471 Issue 10 24 February 2019 Article number 102007 3rd World Multidisciplinary Civil Engineering, Architecture, Urban Planning Symposium WMCAUS 2018 Prague Czech Republic
[2] Nývlt V and Lojda J 2019 Atlantic Press: Proceedings of the International Conference on Economics, Management and Technology in Enterprises 2019 (EMT 2019), The Technical University in Zvolen, Part of series AEBMR Vol. 78 ISSN 2352-5428, ISBN 978-94-6252-714-0, https://doi.org/10.2991/emt-19.2019.25,
[3] Guizzardi G 2005 Ontological Foundations for Structural Conceptual Models Enschede, The Netherlands, 2005, CTIT PhD Thesis Series No. 05-74, Telematica Instituut Fundamental Research Series, No. 015 (TI/FRS/015)
[4] Nývlt V 2018 International Multidisciplinary Scientific GeoConference Surveying Geology and Mining Ecology Management SGEM Vol 18 Issue 5.3 pp 127-134
[5] Lefrancois M and Rasmussen M H 2018 Ontology for Property Management Retrieved from: https://w3c-lbd-cg.github.io/opm
[6] Lefrancois M and Zimmermann A 2018 Custom Datatypes (CDT) Towards a web of Linked Datatype Retrieved from: https://ci.mines-stetienne.fr/lindt/v2/custom_datatypes.html
[7] Schadow G 2017 The Unified Code for Units of Measure (UCUM) Pragmatic Data LLC Clement J. McDonald, National Library of Medicine, Lister Hill, Version: 2.1, Revision: 442, Date: 2017-11-21, Copyright © 1998-2014, Regenstrief Institute, Inc. and the UCUM Organization