Creating semantically valid topic maps

Geir Ove Gr⊘nmo
STEP Infotek A.S., Oslo, Norway
grove@infotek.no
http://www.infotek.no

Abstract:

A topic map, like a database or XML document, provides a way of structuring information and as such requires methods for specifying the rules that describe the allowed states of the map.

This article discusses the need for a mechanism for defining constraints on topic maps to make sure that topic maps are semantically valid according to the intents of the topic map designer. As the paper will show the need for a constraint system is very much there.

Examples of constraints are given in order to give the readers some ideas of how far such a constraint system could be taken.

Existing constraint mechanisms are mentioned and their applicability for describing constraints on topic maps are discussed.

Introduction

The recent publication of the topic maps standard has attracted a lot of interest. This is not surprising since it is a very powerful standard that allows you to create navigational structures that hasn't been possible in a standardized way before.

As people get started with designing their own topic maps they'll soon realize that it is easy to loose the control of its consistency. That is mainly caused by the fact that topic maps easily get rather complex.

Creating and maintaining topic maps without the help of tools introduces the possibility of inconsistency.

Reasonably the standard has only a very limited set of mechanisms for defining constraints on the map. Almost none of those may be used to constrain the semantics defined by the user. The main focus of this article are constraint mechanisms for user defined semantics.

An example of user defined semantics: Associations of type contains must have a container role and a containee role to be meaningful.

Note that this article does not focus on the syntactical representations of topic maps constraint mechanisms.

What are constraints anyway?

Constraints are contracts that are agreed upon by suppliers of a service and receivers of the result of that service. The contract define the conditions under which the services will be provided and a specification of the result of the service that is provided, given that the conditions are fulfilled.

Another definition of topic map constraints could be: **A restriction on one or more of the properties values of nodes in the topic map grove [1] .**

In the context of topic maps this mean that the designer of a topic map ontology [2] and the topic map editor agree upon the rules that define what constitutes a valid and consistent topic map.

Because of the likelyness of inconsistencies the user needs something that helps her to maintain the consistency. Computers are good at this. A computer should be able to guide the user in the right direction and inform her whenever something is incorrect.

Validation

Constraints and validation are related. Validation is merely the act of checking the validity of objects according to a set of constraints. Validation is necessary for all kinds of sophisticated information processing. When designed correctly a constraint system need not be a burden for the user, but would rather guide the user in designing correct topic maps.

Ontologies

All topic maps contain a set of topics that are privileged. These topics are the fundamental semantic building blocks of a topic map and they are the set of topics that other topics and their characteristics are built upon.

When designing topic maps you must give these topics extra attention, since they are so important for how the map is built. As you will soon see the definition of these are also very important to a constraint system.

The set of privileged topics and their characteristics, including associations between them, is what we can call the topic map ontology.

Topic types, association types, occurrence types, facet types, facet value types and themes are examples of ontology topics. One could say that in some sense an ontology contains only abstract topics **-** topics that should only be used as types and themes in a real map.

Type hierarchies can be built by introducing supertype-subtype associations. This makes it possible for ontology topics and derived topics to inherit properties from each other.

Associations between ontology topics can be very powerful, since they can be used by inference engines.

It usually proves very valuable to put much effort into the details of the ontology design.

Designing a topic map ontology can in some ways be compared to defining the elements and attributes for SGML documents.

Roles

[1] ***Roughly a topic map grove can be described as the datastructure resulting from parsing a document conforming to the topic map interchange format. See the HyTime standard for a complete definition of the grove and property set concepts.***

[2] *John F. Sowa gives the following definition in his new book* ***Knowledge Representation: Logical, Philosophical, and Computational Foundations****:* ***Ontology defines the kinds of things that exist in the application domain.***

A short descriptions of the roles that participate in the design and creation processes of a topic map is in place. Two roles are described; the designer and the editor. In practice these two roles may be overlapping or even played by the same person.

The schema design is the responsibility of the topic map **designer**.

The designer should be an expert that knows the domain the ontology is supposed to cover. He must make sure that it is to the greatest extent impossible to create topic maps that are semantically invalid.

When the schema has been designed topic maps that use the ontology can be created. This is where the **editor** takes over.

The editor must use the ontology to create new topic maps and make sure that the topic map objects adhere to the constraints defined for that ontology.

Schemas - the powerful combination

A topic map ontology combined with constraints is what we can call a topic map schema.

This is to some extent the same ingredients as in SGML/XML DTDs and XML schemas. The elements and the attributes define the ontology, while the content models and datatypes define the constraints.

The great thing about schemas is that it can function as the documentation for instances based on that schema. The schema could be all that is needed to understand how topic maps are to be created based on the schema.

Several other interesting possibilities can be derived from this, some of which are:

> It makes it possible to autogenerate user interfaces. A user interface that helps the editor to make sure that invalid topic maps can't be created. The user interface could alternatively be less restrictive and just notify the editor whenever such cases occur.
>
> A suggestive user interface that is able to make inferences about existing topic map information and present this to the editor when useful.
>
> It speeds up creation time, because of the extra help that the editor can get from the tool.

What’s there today?

As mentioned earlier the standard has very little to say about constraints on user semantics. There is almost nothing in the topic maps standard that assists the editor in saying anything about how the objects in your topic map are to be interpreted, less so what incorporates valid or invalid use of them.

It is actually a good thing that no user semantic constraints are defined in the standard, since the number of possible semantics are in practice unlimited. The user semantics of topic maps are dependent on the universe of discourse.

Contrasted to the SGML standard the topic maps standard has no mechanism for applying constraints on ontologies. SGML has a special language for defining constraints on documents called DTDs (Document Type Definitions). There is no such thing defined for topic maps.

A topic map can be serialized in SGML format, but that doesn’t help much. The constraint requirements needed for topic maps are quite different than the ones that are defineable for SGML documents.

The next section discusses the constraint mechanisms that are described in the topic maps standard.

What does the standard have to say about constraints?

The standard is explicit about the fact that it does not constrain the uses to which topic maps can be put. The following note is taken from the section about conformance.:

NOTE 50 This International Standard constrains neither the uses to which topic maps can be put, nor the character of the processing that may be applied by a conforming application. This conformance clause is intended to guarantee that conforming topic maps can be understood to whatever degree conforming read-only applications are intended to understand them, and that the topic mapping information expressed using the topic map syntax will be preserved by conforming read/write applications (except to the extent that the users of read/write applications deliberately alter that information).

The constraints mentioned by the standard apply to topic maps and topic map objects in general. There are no constraints that apply to the semantics defined by the map designer.

The standard defines mostly syntactic constraints. A few semantic constraints are defined as well, but they are on a different level than those that are the focus of this article.

Following is a list of the types of constraints defined by the standard:

The topic naming constraint. Two distinct subjects are not permitted to have the same name characteristic within the exact same scope. It also says that when such a situation occur the topic map application is responsible for merging those topics. Example: Two different topics cannot have the name ’Paris’ in the same scope, e.g. the unconstrained scope. In that case the two Paris topics should be merged by the application, without taking into account that they actually refer to the french capital and Paris in Texas, which surely are two distinct subjects.

Architectural constraints. The architectural DTD include comments, called "conventional comments", that follow conventions established in the HyTime standard to specify syntactic and semantic constraints. For an example have a look at the declaration of the tmdocs attribute on the addthms element form. It says: Constraint: Must be one or more document entities of topic map documents.

Derived architecture. Since the topic map interchange format is defined as an SGML architecture a derived DTD can be created. This makes it possible to explicitly restrict the uses of topic map element and attribute forms. This is however not a very good solution, since not all of the relations are connected through the element hierarchy, but through links. The author of this paper believes that most topic maps will not be developed using the interchange format, but rather using dedicated software tools, which are most likely to have their own persistence system. Thus it is very unlikely that designers will create their own derived DTDs.

Implied constraints. Constraints that are not explicitly specified by the standard, but are implicit because of the fact that SGML is used as the interchange format. Examples: The id of a topic must conform to the restrictions that apply to SGML IDs.

The list of constraints mentioned here is most likely not complete, but the intension is to show the types of constraints described by the standard.

It should also be noted that the standard sometimes is very explicit about the fact that it does **not** limit its uses. An example of this is information resources and their relations to topic map objects:

This International Standard imposes no constraints on the nature of information objects that can be specified as occurrences of topics, nor on the addressing notations used to reference such occurrences.

Which objects are subjects for the constraint mechanism?

Some topic map objects are more suitable subjects for constraints than others. The most important one is definitely the association.

The list of constraint types listed below are not complete. It is primarily intended to be an introduction to which kinds of constraints can be defined. The listed constraints are atomic and are meant to exist in combinations with operators and other constraints.

The examples are chosen arbitrarily, but they can hopefully make things clearer.

Associations

The association type is the primary starting point for describing a constraint for a set of associations. Associations of a given type are very likely to have some strict rules of how those associations should be structured, especially how the roles are be combined.

Associations can be constrained using the following criteria:

- The type of the association.
- The number of association roles.
- The type of association roles.
- The participating topic.
- The type of the participating topic.

Note that the ordering of association roles is not significant.

The goal must be that the association type, association role type and participating topics are combined so that they form a meaningful combination.

Examples

> Associations of type `contains` must have exactly two association roles. One of the roles must be an instance of the type `container` while the other must be an instance of the type `containee`. If the topic that is participating in the `container` role is an instance of `country` then the topic that participates in the `containee` role must be an instance of either `city` or `county`. Otherwise if the topic that participates in the `container` role is an instance of type `county` then the topic that participates in the `containee` role must be an instance of type `city`.
>
> Associations of type `borders-with` must have exactly two association roles. If the association is in the scope `countries` then the participating topics must be instances of `country`. If the association is in the scope `continents` then the participating topics must be instances of `continent`.

Topics

The topic type is the primary starting point for describing a constraint for a set of topics. Topics of a given type are very likely to have strict rules for how the characteristics of those topics are put together.

Topics can be constrained using the following criteria:

> The topic type.
>
> Pattern and length constraints on the identity value.
>
> The number of characteristics by characteristic type.
>
> Valid combinations of characteristic assignments.

Note that the ordering of characteristic assignments is not significant.

Examples

> Topics that are instances of the topic `country` must have exactly one occurrence that is an instance of the type `flag`.
>
> Topics that are instances of the type `country` must have subject identities that match the pattern `"-//Ontopia.net//NONSGML Subject Identity \(Country:[ ]{2}\)//NO"`, where `[ ]{2}` is the two-letter country datacode. Such pattern matching would be useful if you wanted your topic map to have a consistent use of subject identities.
>
> Topics with names in the scope `norwegian` must have at least one occurrence of type `description` in the scope `norwegian`.
>
> A topic that is an instance of the type `country` must participate in exactly one binary association of type `contains`, where the other participating topic is an instance of `continent`.

Names

Topic names are containers for sets of base names, display names and sort names.

The base, display and sort names are basically strings, so they can be constrained in the same way as any other string.

Name can therefore be constrained using the following criteria:

> The valid combination of base names, display names and sort names.
>
> Patterns for matching the name strings.
>
> Name length.

Examples

Topics that are instances of the type `country` must have at least one topic name in each of the scopes `fullname`, `local-name` and `datacode`. Topic names in the scope `fullname` must have exactly one base name and one sort name.

Occurrences

Occurrences could be restricted by what kind of information resources they are locating and how those resources are addressed.

The notation of the information resource (e.g. GIF, HTML).

The resource location (e.g. in-house, located in a specific country).

The addressing notation (e.g. XLink, HyTime).

The address type (e.g. URL, nameloc).

The properties of the address (e.g. by protocol: http, ftp).

Examples

Occurrences that are instance of the type `flag` in the scope `cia-world-factbook` must have addresses that are URLs pointing to jpg files located under `http://www.odci.gov/cia/publications/factbook/`

Occurrences that are instances of the type `description` must point to information resources that are XML documents.

Scopes

All topic characteristics; assocations, names and occurrences, are subjects to be constrained by their scope.

Scopeable objects can be constrainted using the following criteria:

The set of valid themes for a scopable object.

Topics that are allowed to be used as themes.

Themes that must be used together.

Themes that must not be used together.

Examples

Themes that are instances of the same topic type must not be used as themes for the same topic characteristic. E.g. the themes `norwegian` and `english`, which both are instances of the type `language`, must not both be used as themes for an occurrence that is an instance of the type `description`.

A more specific variant: All occurrences that are instances of the type `mention` must be scoped by exactly one theme that is an instance of the type `language`.

Only topics that are instances of the type `theme` can be used as themes.

Other types of constraints

Vocabulary based constraints, where the set of valid values of a property must or must not be members of a predefined vocabulary.

Other constraint mechanisms

The following constraint mechanisms are some of the contestants for becoming a mechanism for specifying constraints on topic maps:

The Object Constraint Language - an OMG standard for specifying invariants, preconditions, postconditions and other kinds of constraints on UML models.

EXPRESS - a data modeling language that is part of the STEP standard.

Topic Map object model API - a (currently non-existing) programming language API. The idea is that an API could be created for topic maps in the same way as they are for XML through the Document Object Model (DOM). This API should preferably be programming language independent, e.g. described using IDL.

A generic property set / grove constraint language which is able to work with any grove. Unfortunately nobody has invented this yet, but it has the potensial of being the most powerful and flexible constraint system that could ever be invented, mainly because it would be totally independent of the data notation.

The Topic Maps standard - topic map instances can be used for describing constraints.

Note that XML schemas is not on this list, since it does not give support for the kinds of constraint mechanisms needed for topic maps. This critisism is mainly directed at its lack of inter object constraints.

How to apply constraints?

The validation procedure normally consists of two steps. These are:

1. Decide which objects are qualified as subjects for the constraint.
2. Apply the constraint on the qualified objects.

How to best describe constraints?

The author's experiences indicate that a complete programming language is needed to describe the possible constraints that could be needed to describe all kinds of constraints on topic maps.

The most flexible solution is to use an existing programming language to describe constraints in terms of the object model api of the topic map system.

Even though this would mean that a programming language was involved, user interfaces would easily be created on top of that, so that anybody could be able to describe constraints, not just programmers.

80/20 solutions can be created by removing the most complex constraint requirements. This would make the language much simpler, and it would no longer have to be a complete programming language.

Early experiments show that topic maps themselves can be used for creating a constraint mechanism. This mechanism is based on a templating technique that constrain the instances of those templates.

Conclusion

A constraint system that is able to define constraints on topic maps are sorely needed. Before deploying real world topic maps it is very useful, if not necessary, to be able to make sure that its semantics really are the same as the ones you intended it to have.

Let's hope that the topic map community is able to come together to agree upon a language for defining constraints. Without it we would end up with a lot of different and incompatible languages. That would be very unfortunate and probably limit the interchangeability of topic maps.

Author

Geir Ove Gr⊘nmo

Information architect
STEP Infotek A.S.
Postal Address:
Gjerdrumsvei 12
0486 Oslo
Norway
Telephon: +47 22 02 16 84
Fax: +47 22 02 16 81
E-mail: grove@infotek.no
Web: www.infotek.no

Geir Ove Gr⊘nmo - Geir Ove Gr⊘nmo is an Information Architect at STEP Infotek, a Norwegian company that specializes in information reengineering. He studied Computer Science and Graphic Arts technologies at the College of Gj⊘vik and joined Falch Infotek, now STEP Infotek, in 1995.

During his military service Geir Ove worked with the implementation of SGML technologies at the Norwegian CALS office. He is currently involved in the design and implementation of document management and Topic Map solutions. He has been involved in a number of other projects in which SGML, HyTime and DSSSL technologies have played a key role, and is the author of several Open Source tools for processing topic maps, groves and SGML architectures, including tmproc, GPS and xmlarch.