Topic maps

Author: Michel Biezunski
22 September 1998
Last revised:December 17, 2002

Business Introduction to Topic maps

Conceptual introduction to Topic maps

General presentation

Grass versus flowers: the nectar of information

Today's World Wide Web is like a meadow that would cover the planet. A gigantic ocean of unidentified grass, available for browsing. Its potential is enormous, but there is no indication on how much time it takes before finding a specific piece of information. The main reason is that it is difficult, albeit impossible, to identify any semantic information on the Web. The way it works basically enables us to go from a world-wide meadows to smaller meadows, until we find one we are interested in, and then we start browsing it until we find the blade of grass we really want.

Before the Internet existed, browsing was an activity mainly performed by cows. Cows eat grass, and spend their life chewing it again and again. Grass is desperately green. No obvious sign distinguishes one blade of grass from another. Looking for one piece of information can be like looking for a blade of grass within a meadow. We know it is there, it just takes for ever before finding exactly the one we want. Flowers, on the contrary, are easy to distinguish. They have a characteristic color, shape, and smell. And bees, contrarily to cows which are just chewing grass, gather nectar on flowers, which they not only use it for internal consumption, but become vectors for fecundating new flowers due to transfer of their seeds to other locations. Bees also are working hard. They build bee-nests and produce honey.

Flowers can be cultivated, or wild. Like information repositories, which can be structured or unstructured. When cultivated, flowers are easy to group by types, families, colors, and they form well-ordered fields of different colors, or harmoniously designed gardens. Wild flowers, on the contrary, are more difficult to identify, although they florish in the country, in the mountains, and even in the cities.

Whether wild or cultivated, flowers are better than grass blades. They have the characteristics that help locating and differentiating them quickly and efficiently.

Finding flowers in cultivated areas which are specialized in flowers of a certain type is relatively easy. What is more challenging is finding the wild species that represent sometimes the real value of what we are looking for.

We need to find flowers not only for browsing, but for gathering their nectar, in order to produce new flowers and honey. We want to be bees, because cows are no more a suitable model.

This is what topic maps are about. They are the nectar of information.

The Infoglut

Since the early 1980s, when computers have started to be used by everyone and have become common on every desk, the nature of information and the way it is used has changed dramatically. The first generation of word processors and desktop publishing packages was based on the WYSIWYG concept. "What you see is what you get" implies that what you get is a printed result. At that time, computers were primarily used as sophisticated typewriters to produce information for printing. Paper was the major means of interchanging information.

Since the beginning of the 1990s, the situation has changed. The big move has been represented by the World Wide Web. It became a forum where everybody can make his own information available to the entire world and read information produced by everyone else, with minimal cost and generalized access. The Internet revolution changed the entire world, by reducing geographic distances, removing political barriers between states, and the emerging world-wide commerce that is starting obliges states and governments to revise their fundations, legal, fiscal agreements, trade treaties, copyright laws, etc.

Since the emergence of the World-Wide Web, we are producing information that is not intended for being printed. What is essential instead is that links are able to lead to other appropriate information objects. Instead of producing huge quantities of information, such as the one that is stored in collections of books, we are now splitting it into small chunks, called "Web pages", linked one to another and allowing quasi infinite ways of access. These web pages may contain meta-information used by search engines to return lists of pages containing the word or phrase we are looking for. Because the technologies built for browsing the Web have become so widespread and therefore very accessible, the ways of organizing and navigating information initiated by the WWW has spread out th other areas of information management, and in particular the huge information repositories stored and maintained within companies. The concept of "Intranet" means that what works for the Web also works for local information.

Because there is no limit to what can be put on the Web (at least in the democratic countries), the Web is growing at vertigineous speed, and contains millions of pages. But all this information is not useful. The volumes of information made available are so huge than the challenge now becomes: how can I easily find the information I really need? There is a universal need for describing how information can be efficiently retrieved. And this is precisely why topic maps have been invented.

When books have been first produced, libraries arose as ways to store and preserve them, and (sometimes) make them available to the public. Librarians have started to organize catalogs, sorted by authors and subjects. Navigational devices have been invented for each document, to facilitate locate location of relevant information: tables of contents, indexes, glossaries. Thesauri are multi-document devices, that help finding terms that can in turn be used to search where information sources can be found.

Tables of contents, indexes and glossaries can of course be used also for electronic documents. However, these devices have been designed with the printed paradigm in mind and are far from optimized for online documents. For example, tables of contents should generally be derivable automatically from the hierarchy of containers within a document, or from a list of titles. Indexes are more complex, because they reflect how the author(s) or the editor(s) intend information to be retrieved. Some indexes add considerable value to the information set they refer to, and are quite complex intellectual constructions, difficult to describe in a general way. The Topic Map Model aims at describing ways indexes are designed, even in a quite complex way. It is clear that the typography used for indexes, such as indentation for subentries, or use of bold characters for some references, are a mere formatting artefact behind a hidden structure, which is much more complex than just a tree of entries and subentries. awkward to use and their design logics is far from obvious. Glossaries are limited to one book, and are often inconsistent over multiple books, even on the same subject. Cross-references are very useful, but very time-consuming to create, and difficult to maintain when documents evolve. Hence the need for new navigational devices optimized for navigation.

What are topic maps?

- A device for organizing navigation within information.
- The electronic equivalent of indexes in printed books
- Plus glossaries, thesauri and cross-references
- A management tool for living documents
- The possibility of creating easy, intuitive user interfaces
- An internationally-standardized, interchangeable representation.

Topic maps are perspectives on information. They express views in which some set of information objects can be seen. They allow multiple navigation strategies to be fully explicitly defined. Therefore, they enrich the value of the information set to which they are applied, and they allow their authors to deliver information that can be customized at will to fit specified user needs.

Topic maps are devices to organize navigation within information. Moreover, they represent a way to structure information, because organizing navigation means applying a semantically-based structure on existing information. This structure may rely on pre-existing structure, or can be added on non-structured information.

Information repositories that have addressable structure belong to two categories: databases and structured documents (XML/SGML). Information repositories that do not have addressable structure include text files, word processors or desktop-publishing produced documents, videos, films, music, etc. Some information objects might be hybrid: for example, some documents include a header which contains structured meta-data, followed by a body which is unstructured.

Topic maps can be used to merge information objects that have structure with information objects that do not have structure. The difference between these two forms of information vanishes once a topic map has been applied. Building a topic map by retrieving a pre-existing structure can be automatic, while building it on top of unstructured information objects requires other processes, such as natural language processors, automatic search engines, and/or manual work. However, once the topic map exists, the fact there was structure to begin with in the source documents is no more relevant. Topic maps can also be used to alter the original structure if necessary.

The reason why this powerful feature works is the fact that topic maps apply from outside, and can be applied to any kind of information. Topic maps are unobstrusive views over information. Furthermore, several topic maps can be superimposed on the same set of information, therefore allowing customized views on information.

Topic maps answer the need for managing large information repositories, that are constantly evolving, and where relevant information needs to be found quickly.

Maintaining links is not an easy task when documents evolve constantly. This remark applies to technical documentation, dictionaries, etc., as well as to Web sites. The main problem for management comes from the fact that links are usually perceived as uni-directional, because this is the way they are used on the Web. They start from an origin and go to a target. When the target changes, then there is no easy way to report automatically these changes to the origin of links. Therefore, links become obsolete. To maintain links, it is necessary to create specific software applications, that are able to check whether the targets still exist. If they don't exist any more, they have either to be re-created, or links have to be removed.

The work of creating and maintaining link integrity is generally time-consuming, and painful. Even on the Web, it is still performed manually most of the time. It is necessary to insert the target of the link for each origin of the link. The "point and click" type of interface helps, but it still takes much time to create each link. And link integrity is not preserved using this method: each time link targets are modified, this work needs to be performed again.

These types of links, also known as "cross-references" in the printing world, do not carry any semantics. They are called "contextual links" because the reason why the link exists can be understood from the context of the origin of the link. If the origin and the target are both called*anchors* of the link, then a link is contextual if one of the anchors of the link (the origin) is the link itself. Contextual links basically contain only the address of the target.

There are two aspects in information links: the addresses of the information objects connected together (the anchors) and the reason why these objects are connected (the roles played by each group of anchors). Setting the addresses of information objects is something machines can do, and which is time-consuming and not interesting for humans. This activity is like reporting lists of page numbers for each index entry in printed books. It is boring, while still demanding for highly qualified people, is subject to errors, and is not sharable by even only two different persons. Setting the semantics of a link is something machines have difficulty to do, because it is necessary to have a global knowledge of the context in which the link is made to decide the kind of relationship actually expressed by a given link. But this is an activity which is interesting for humans to do, because this is where the real value

of information is, and this is what will make information really usable or profitable and therefore will give a competitive advantage to its owners.

Building a topic map basically means that instead of having to create links by hands, one has to qualify what the information object is about. This activity results in the creation of topics. Where topics are exactly anchored is then a task that can be left to the machines.

It is not an easy task to define what the relevant topics are applied to a given set of information. But when this effort has been accomplished, then one would find highly desirable to be able to preserve all the information about topics and their relationships, in order to maintain, extend, and merge several topic maps, while having access to a variety of software applications able to understand and process topic maps.

This is now made possible, as topic maps are an international standard. ISO/IEC 13250 answers the need for interchanging topic maps in many different environments. The standard does not define any specific semantics for topic maps. The types and instances of topics, their relationships, scopes, and their multiple facets are left to the creativity of each user or user community. The standard only provides a template, a set of placeholders. Topic maps sophistication can occupy a full range from very simple to very elaborated. Their characteristics depend on the imagination and requirements of their designers.

The limits of computer-aided knowledge management

Communication between humans work because language is ambiguous. There are things that get transmitted from parents to children, or from teachers to students without having been formulated. They "float around". What we really, intimately, communicate is not always something we need to say. We communicate also with the eyes, with our body, with our gestures, and sometimes just by our presence. Computers are not very good at that. They need to process information which has been put in a form where ambiguity can't be tolerated (even if recent development on fuzzy logics and related issues show that it's possible to go further in that direction).

Topic maps are by nature a limited representation of knowledge, because they are really useful when they can be used to feed computers. But then, they are really usable.

To illustrate this point, let's take an example, where somebody has carefully prepared a topic map, in which items have been assigned a color with a name. This topic map is merged with another one, which also deals with colors. At the moment of the merge, terrible results are produced, because in the first topic map, there are three nuances for "blue": Light blue, Dark blue, and Blue, and in the second topic map, there are one hundred different nuances for blue. Therefore, the nuance called "light blue" in the first topic map might correspond to 20 or more colors in the second topic map. In other words, the granularity of the representation of knowledge is different, and therefore the mapping is not easy. Still, it is possible to say that each color in the 2nd topic map corresponds to 20 colors in the 1st topic map.

This limitation does not mean that topic maps are useless. On the contrary, it means they are not only useful, but powerful. Because they help us represent our information in a quite sophisticated way, with multiple qualifiers serving various purposes: the information itself, the role played by specific occurrences of it, the type to which it belongs, the scope within which it is valid, and the multiple facets it may exhibit.

Agreement on semantic granularity is therefore a requirement. "Public topics", as defined in the Topic Maps international standard, can help designing and document common layers of shared metadata. An industry, as a whole, can greatly benefit of information interchanges based on topic maps. The fact to know where are the limits of its applicability enables at the same time to know the extent of power it has.

Merging the Document World and the Database World

Computers help us deal with information. There are traditionally two ways to use computers: well structured information sets, usually called databases, and free-form information, intended for human readibility: graphics, and documents.

The SGML approach, now pursued under the XML umbrella, represents a mixture of both worlds: information is structured, and is split into significant portions (called "elements"). The representation of information (style rendering), and the various processes that can be performed on it (sorting, extracting, and in general transforming) are greatly facilitated. The big industry knows already the advantages that such an approach has on information management.

XML proponents aim at widening SGML acceptance. By removing the heaviest, and therefore most expensive, features, XML makes SGML affoardable to the masses. And therefore it becomes possible to extract, transform and interchange chunks of structured information on the Web -- and elsewhere.

XML goes one step further on the way between documents and databases.

Topic maps play a role in this process in going one step further. Topic maps are offering a way to create generalized external meta-data mechanisms.Information does not have to be previously structured, it can be used as if it were structured. In other terms, topic maps allow us to do external structuring. Aftermath is possible, and therefore the flexibility is greatly increased when merging existing information (archive in various formats) with brand new one.

The publishing paradigm

Table of contents with Topic maps

A table of contents is generally made of a list of headers used as chapter titles, section titles, etc., down to a given level of subtitles.

Most word processing software applications enable users to automatically create tables of contents from such elements, usually marked up with bookmarks or special styles in predefined style sheets.

In other words, a table of contents can be interpreted as a partial view of the documents, where all information but the headers is hidden. Tables of contents entries respect the order in which the document has been created.

Topic maps have no particular feature to add to tables of contents, except that, like indexes, thesauri or glossaries (see below) they can be considered a resolved query in a specific database.

Indexes with Topic maps

- Indexes are viewed as lists of topics sorted by topic types.
- Same functions as traditional indexes (sometimes differentiated between e.g., index of names, index of concepts, index of locations, etc.)
- Topic maps provide semantics of occurrence roles. Equivalent of bolded page numbers in printed indexes, except that semantics is explicitely declared an no limit to the number of different semantic values can be applied.
- Topic maps provide bidirectional navigation: from an index entry to a topic occurrence, and from a topic occurrence to the index entry.

The classical notion of a printed index changes meaning when dealing with online documents. Indexes can be considered essentially as navigational devices, i.e. tools to access directly to a specific information within a set of data objects. An online index is then different from an ordered sequence of entries and subentries followed by page numbers.

The origin of the Topic Map model is the quest for generation and maintenance of multiple document indexes. Unix systems vendors were looking for ways to improve consistency between the documentation of the systems created by one given company,

third-party companies, and the books written on the same subjects by independent authors.

Printed indexes look like hierarchical structures made of entries and subentries, down to a given sublevel, followed by lists of page numbers (or section numbers) indicating where the occurrences of given concepts are to be found within the document. But a depper analysis on indexes shows that this hierarchical structure is an artefact, caused by formatting constraints due to paper/printed technology. Entries and subentries are far from representing a real hierarchical relationship between concepts. It often happens that a subentry in an index occurs as an entry in the same index, and vice-versa. The author of an index indicates thereby that there is some kind of relationship between the entry and the subentry, and tells the reader to also look for another topic related to the main entry, even when the relationship is not strictly hierarchical. The fact that this proximity can be repeated several times in an index is due to the fact that multiple internal cross-references within an index are not convenient. Therefore, a good index accepts redundancies to improve convenience. Sometimes, though, cross-references do exist within an index, an appear with the indication "See" or "See also" another entry. Usually, when "see" is used, it means that the reader must go first to another entry before she can see the list of occurrence addresses.

Building and designing an index which is useful is an intellectual creation. The logical rules behind its design are very complex, and most of the times not explicitely stated. A human indexer proposes a program for reading the book by abstracting the key concepts and offering navigation that provides the most convenient access to the main topics in the book. The complexity of an index is reflected by the fact that the process of creating an index is very difficult to transmit. If an indexer interrupts his or her work before completion, and the index has to be completed by another person, chances are that the resulting index will be different, compared to what it would have been if the first indexer would have been able to fullfill the work. In other words, an index is a human creation, that provides a perspective on the work being indexed.

Browsing a printed index gives an overview on the work. Entries that are followed by more references than other are likely to weight more, although this is not always the case. Some of the key concepts are mentioned only where the main focus is, because too many references for an entry become useless. There is a limit where too much information becomes like no information. When browsing the Web, for example, if there are more than several dozens of hits, it is usually difficult to find relevant information, or very time-consuming.

In many indexes, some references are printed in a font variant, such as bold or italics, indicating that the occurrence to be found at this location plays a specific role. It is usually the main reference of the concept. In general, the meaning of this typographical variant is explained in a footnote at the beginning or at the end of index.

Several indexes may be present in the same book or collection: subject index, index of names, index of locations, etc. When only one index is present, it can be limited to a subject index, or include different entry types, all mixed together.

While traditional indexes are difficult to create, they are even more difficult, when not impossible, to maintain, when documents evolve. It it then necessary to read all the parts that have been modified and check whether they contain occurrences of existing entries, and it is necessary to check practically for any existing entry if the reference still exists and is still correct. There are computer-based tools to help doing this work. But it is so difficult to come back and enrich an existing index, that sometimes it is considered easier to start from scratch. This work is also often considered too big and time-consuming to be profitable.

Tools exist to create printed indexes with a computer. Most word processors and desktop publishing software packages have a feature letting users insert markers for index entries in the document, and often provide the possibility of declaring subentries. An alternative or complementary way to do it is for the user to provide a list of strings that will be used to mark occurrences automatically within the document, or set of documents. Once this work of inserting index markers is completed, an automatic indexing processing takes place by collecting all these markers and presenting them in alphabetic order, followed by page numbers. The references can alternatively be expressed as hypertext links, allowing for online navigation.

During the early phase of the development of the topic map concepts, we have felt the necessity to re-conceptualize what the united model of an index is.

Because an index entry is made of both the string representing the concept and the addresses (usually, page numbers) of its occurrences, it can be mapped to what has been conceptualized in HyTime as a link. Furthermore, it is an independent link, i.e., a link that can be expressed outside of each location that it is connecting together. Having recognized that feature, then it is enlightening to apply the major breakthrough brought for by HyTime, i.e., the conceptual difference between location and linking. The location part is the list of addresses to which this link point, which can be either the list of page numbers, or the list of addresses expresssed as targes of hypertext links. The linking part is the semantics of the relationship expressed by the link. In a printed index, the semantics is very limited, and usually not explicti. A page number following an entry in an index usually means: there is something*about* this given topic at this location. In other words, it is a*mention* of the topic. When a typographical variant is used, such as bold for page numbers, the semantics is most of the times made explicit, such as :*Main mentions* of the topics are printed in bold.

Designing the Topic maps view of a printed index has gone through a first step of making the semantics which is either not expressed, or limited, explicit. Therefore, a topic as defined by the topic map belongs to a type: it can be a term, a person name, a location, etc. It is left to the Topic Map designer to state how many different categories of topics are in use, and what these categories mean. It is also a design decision to differentiate between occurrence roles: "mention" and "main mention" (bold in an index) are only a sepcific case. It is possible to provide an access as precise as desired to specific occurrences. For example, it can be useful to indicate that an occurrence of a given term is a graphic or a definition. Rather than providing a list of figures, and have the user search to try to locate a figure relevant to the topic of interest, it is possible to relate one given topic with the figure(s) that are relevant to it. This inversion of usual practice illustrates that Topic Map design priviledges semantics over physical appearance of occurrences, such as figures.

Designing an index in the context of Topic maps includes the following procedures:

1. Define topic categories
2. For each category of topic, differentiate the occurrence types.
3. Create topic instances (by giving them a name)
4. For each topic instance, and each occurrence type, locate the occurrences.

At first sight, creating an index with a topic map seems harder than creating a traditional index. In an equivalent context -- limited number of topic types -- the work is basically the same. There is more work if the precision of the topic map is higher than the one of the index. However, many steps in that procedure can be automated. Some information may be extracted automatically, such as metadata that may exist in a document, or data that can be retrieved because it is possible to locate the context in which it appears. Search algorithms are applicable, an can be as sophisticated as desired. The result of hits is made into a form (document or database) that can be controlled by the author of the topic map. It is therefore possible to edit and refine a topic map.

For the same reason, an index made with a topic map is easy to maintain. If a new data object is added, or if portions of documents are added, the only thing that needs to be done is to add a pointer from the topic to the location serving as an occurrence for this topic. Symmetrically, where an information object is deleted, the only thing to be changed is the pointer from the topic to the location which has been deleted. In both cases, the remaining addresses for the topic occurrences rearrange themselves automatically.

One of the leading principles in the Topic Map design is to keep it as simple as possible. Therefore, there are no notions of subentries for the topic. However, the same functionality may be achieved by creating another topic and asociating it to the first. Each time there is something to say about a thing, this thing becomes a topic.

Associating topics together is possible using "association" links. The semantics of the association is left at the initiative of the Topic Map designer.

An entry is associated with subentries with an association which realizes the semantic of containment. This link has two anchors: "container" and "containee". In a general case, the anchor "container" will contain a pointer to one object (a topic) and the "containee" anchor may contain a whole bunch of topics, grouped in an aggregate.

Entry-subentry relationship semantics may be different from simple containment. In indexes, when preposition such as "at", or "as", are used, it implied the semantic "located" and "playing the role of". These semantics can be explicitely defined and differentiated in a topic map based index.

Because associations are described as links, there are no constraints such as strictly hierarchical containment. A topic may be contained in several other topics, without any relationship with the hierarchical level in which other topics are located, in the "containment tree". This offers the maximal flexibility and helps users define models that can be fairly sophisticated and well suited to their needs while remaining intrincsically very simple, from the point of view of the underlying architecture.

Bidirectional navigation

Glossaries

- Definitions are considered particular occurrences of a topic, with the value of "occurrence role" being definition.
- No need to create separate glossaries. A glossary can be created by assembling the portions of information that are qualified as definitions.
- Definition of topics are linked with other occurrences through the topic construct.
- Several definitions may coexist for a given topic.
- Definitions can occur in several languages for the same topic (and can be filtered).
- Topic maps provide bidirectional navigation: from a definition of a term to the other occurrences of this term, and from any occurrence of a term to its definition (if any).

A glossary is generally a list of terms followed by their definitions. There is some redundancy between the glossary and the index. Terms used as glossary entries are often found as well as index entries.

Sometimes, both are mixed together and the index contains the definition of the term followed by the list of their occurrences. In that case, the index is called a "master index", or "index-glossary".

Now, it is possible to consider term definitions as specific occurrences of the term. Therefore, a glossary would be like an index that, instead of containing pointers to any mention, contain instead pointers to definitions.

This remark can be extended by noting that, rather to create a separate glossary, it is often more convenient to report that there are definitions of the term interspersed somewhere within the data repository, and it really should not matter whether the definition is to be found in a specific document called a glossary or anywhere else.

Furthermore, it is not necessarily the case that the term definition is limited to one or two paragraphs. Sometimes, the definition of a term can be the subject of a whole document. It can also happen that a term is defined more than once, and then it may be interesting to give users the access to all existing definitions, that may or may not be differentiated by their scope.

This architecture facilitates maintenance of the glossaries. If a definition is changed, but its address remains unchanged, the topic map automatically gives access to the new content of the definition. It also contributes to facilitate navigation. It enables us to go from any occurrence of a topic to its definition, without having to repeat the definition everywhere. This hold as well for restricted views of definition, for example, the one that links an acronym with its extension.

As a matter of fact, what is said about definition holds for any other occurrence role. In the case of a topic category called "person", "contact information", or "biography", can be considered playing the same role as definitions. A "glossary" of "contact information", i.e. an address book, is then the same as a glossary, viewed from a Topic Map perspective. Both are lists of topics in alphabetic order, followed by the content of the anchors corresponding to a given role.

Cross-references and Bibliographic references

Cross-references are actually links between two (or more) occurrences of a given topic.

- Cross-references are "degenerated" topics, because they usually lack the semantics of why there is another occurrence related.
- As cross-references are "burned" into documents, they are very difficult to maintain once anything changes in a document.
- Web links are basically the same as cross-references.
- Cross-reference maintenance can be the first motivation to adopt topic maps.

A cross-reference is a navigational device used to trigger traversal to another portion of the same document.

A bibliographic reference is basically the same, except that it points to an external document, either to the document as a whole, or to a specific location within the external document.

The notion of what is "in" and what is "out" the document may vary in time for the same reference. Within traditional navigational devices, such as printed books, a cross-reference is pointing to a page that happens to be inside the very same book, while a bibliographic reference indicates either the title of another book and article, possibly followed by a page number inside this second object.

Traditional SGML applications follow the book paradigm. Several "subdocuments" are assembled in a main "document" which is considered as a book. Therefore, the notion of "internal" versus "external" still makes sense. In hypermedia applications, links are made indifferently to another location in the same document (sometimes limited to a small Web page) or to other documents.On the World Wide Web, for example, links are made to other Web pages that might be located in other companies' pages, sometimes physically located at the outskirts of the world, but still accessible as if all files would be residing on the same computer. In this kind of applications, the difference between external and internal is blurred, because the user doesn't see all the processes going on to switch between pages, networks, countries.

There are cases where references can not be accessed. One case is when the target doesn't exist in electronic form: it is a book that has to be ordered in a library, for example. Another case is when the target is owned by a different owner from the source, and the owner has set up restrictions to access his materials. In all these cases, references are called "bibliographic references" instead of "cross-references".

Cross-references mostly differ from bibliographic references by the fact that occurrences of cross-references are part of the information space on which the topic map applies, while occurrences of bibliographic references are outside the border of this information space. This difference does not change their referencing nature, but changes the kinds of processing used to access them and to navigate from them. A bibliographic reference can only display the reference, while a cross-reference can trigger traversal and lead to the actual information object used as the occurrence.

Because topic maps are based on HyTime, which contains various addressing mechanisms, the notion of "in" and "out" is different from the traditional case . On the one hand, what is "in"is what is directly accessible from the origin. Even if the information that is pointed to is located on a remote Web site, it is still considered "in" if it's accessible from the site. On the other hand, it is considered "out", if it is not directly accessible, for example because it does not exist on line. Therefore,k the notion of cross-reference in this context is more extended than in the traditional case. A bibliographic reference is defined as not accessible electronically.

A cross-reference has a source and a target. It is generally displayed with "see".

Thesauri

- Topic maps allow flexible thesauri to be built
- The links are not limited to a hierarchical structure. They can represent graphs instead.
- As links are made with topics, it is possible to go from one topic to all topics related, and from each topic or related topics to its occurrences.
- Therefore, each occurrence of a topic in the document is automatically connected with the thesaurus.

The Topic Map model is well suited for the design and flexible and powerful thesauri, and/or integration of existing ones.

The term "thesaurus" has several meanings. Thesauri can be considered as a set of terms related by specified relations. Within the topic map architecture perspective, the terms are considered topics, and the relationships are described with associations. Since the Topic Map standard does not enforce any specific semantics for the associations, topic map designers, including thesaurus designers and users, are free to define their number and the semantics they want when instantiating the associations. For example, in the ISO 2788:1986 standard for the creation of monolingual thesauri, three types of relationships are allowed: equivalence, hierarchical, and association. These three types can be directly described as instances of topic map associations.

The constraint introduced by the Topic Map model is that only topics can be associated together. Therefore, to relate a "thing" with another thing, these two things need to be considered as topics. The fact that these topics do not point to occurrences within the information sources doesn't prevent considering them as topics. They can be called "hanging topics". But it is possible to first create them because they belong to a thesaurus, and in a second step consider them as index entries, and relate them with the information objects on which the topic map is applied.

The advantage of the fact that associations only occur between topics, is that they are independent from the information objects to which they apply. This has the consequence that the same thesaurus of relations can be applied to a variety of documents or data objects in general. The work of creating the thesaurus can be done independently as well and can be shared by a community of users, for example a whole industry.

The Topic Map architecture is open. Therefore, it doesn't solve the problems or ambiguities that can result from the design of a thesaurus. It provides a placeholder for the description of existing or future thesauri, and it provides the maximum flexibility, because the relationships are not only hierarchical, but also graph-oriented. Therefore, it offers opportunities for later improvement and sophistication of a thesaurus-based application.

The Topic Map architecture does not constrain the nature of the things that are linked together. Within the constraint that only topics can be related together using associations, there is still much freedom. If a topic map designer needs to constraint the nature of things that should be associated, for example by saying that only certain categories of topics can be related through a given association, it's possible to say so, by adding reference typing constraints in the document type definition describing the Topic Map.

Thesauri are always created by Topic maps, just by virtue of the fact that they are at least a list of topics. The rules that apply to relate topics together is under the control of the user. The thesaurus is automatically connected to the indexes, and to the glossaries.

In a library, it means that a term catalog can be unified with individual indexes of books, and therefore can enable access to the final destination resources, suc as a given paragraph, or a graphic in a given book or article. There are no conceptual limits on what can be unified. Retroftting existing information to allow universal linking at any possible level of granularity would be an immense task and is practically unfeasible. But doing so by previously planning it with information yet to be created is now perfectly feasible.

Topic maps and the Web

Improving Information Retrieval

Push Technologies with Topic maps

- Topic typing can help information providers sort their information.
- Filters can help differentiating further on orthogonal criteria.
- Several filters can be used concurrently to further differentiate the information to be delivered.
- Fine adjustment is therefore possible.

The Post-structuring concept

Placeholders for user-definable semantics

The model defines templates that topic map designers can use to describe their navigation model and filtering strategy. Each of those corresponds to a different aspect, well identified, of the description of information. Furthermore, these place hoders rely on emphasizing the difference between semantics (anchor roles) and addressing, the whole area of addressing being excluded from the scope of Topic maps because reference is made to HyTime (subsets of addressing mechanisms will differentiate specific implementations).

An Information Overlay

Meta-meta-data

The Post-Structuring Concept

When information is structured, with a database, or with XML/SGML tags, there is a model, either explicit or implicit, which describes the structure of information. In SGML, the structuring model is expressed using a syntax defined in the ISO 8879 standard, namely the "document type definition" formalism. In XML, the model is expressed either with a similar, albeit slightly simplified representation, or with a schema language (still to be agreed upon). However, XML doesn't impose this model to be present. Actually SGML does not impose either this model to be explicitely defined, since the modification known as "Web SGML" which has passed in

1997.

Regardless of the fact that the model is explicitely defined, tags are used to structure the information. This situation is opposed to the one where only formatting markup exists within documents. The markup produced by word processors is an example of such markup. It is, semantically speaking, unstructured.

Web editors represent a mixture of both. HTML elements are tags which resemble very much to the SGML/XML tags, but do not bear any semantics that can usefully be taken into account for navigation. Therefore they are more like the markup used in word processors or desktop publishing software applications. And this is even more so since there exists add-ons for dynamic features, and complex formatting.

Navigation, on the other hand, requires locating information relevant to given subjects. Therefore, in order to optimize navigation, it is important to locate where things relevant to given subjects are located.

Topic maps apply as well to unstructured and structured documents.

The model defines a general indirection mechanism which says basically: if you don't have the information you need within your existing documents, it's not too late, here is an opportunity to add what's missing from outside by using links. And, if you happen to have already the information you need, then you just point to what you have, building a Topic Map will be automatic.

ISO Standard Explained

ISO 13250, Topic maps. Scope and Schedule

Scope

ISO 13250, "Topic Maps", provides a syntax based on a model aimed at defining overlays on existing information. These overlays provide views, also called "topic maps". There can be multiple concurrent views on the same set of information objects.

The model includes three components: *Topics* are used to connect all information objects relevant to a given concept, *associations* are used to relate topics together, and *facets* are used to assign supplementary properties for other purposes than purely topic navigation.

These three constructs are all expressed in terms of "independent links".

The TNM standard is limited to express the information necessary to build topic map navigation. It provides no specific mechanism neither how to create and process topic maps, nor to display them. Application designers are free to conceive any way they want to build and exploit topic map information.

The kinds of navigation that are enabled using Topic maps is the electronic equivalent to indexes, glossaries, thesauri, cross-references, tables of contents and catalogs, used as basic navigation mechanisms and originating from the world of printing.

The fact that such an international standard exists makes it possible to interchange navigational strategies and components, i.e., the link databases that have been created to help navigate within an information set, and provide views on this information. For example, industry-wide lexicons, lists of authority keywords, indexes, glossaries, can be sold and plugged into several application. It also makes it possible to relate huge information bases made by independent authors, such as competitive companies, as well as to offer a common interface to the user of Internet, for example for Web-based Electronic Commerce applications, or for relating library catalogs together, even if they have been built using incompatible classification schemes.

Schedule

- ISO 13250 Topic Maps, due in February 1999.
- Under the auspices of Hypermedia Languages Special Working Group of ISO/JTC1 WG4
- Co-editors: Michel Biezunski, Martin Bryan and Steven R. Newcomb.
- First Committee Draft issued in May 1996 (= CApH Topic Map Spec.)
- Proposed Simplification: January 1998 (with addition of filters)
- Revised draft in may 1998
- Final CD produced in the fall 1998
- IS in December 1999.

Associated Standards: SGML, HyTime, XML

SGML is the standard in which all others used here originate. SGML provides a powerful foundation for describing information, by separating structure from content. Structure is described by generic markup, i.e. sets of tags that are user-defined. SGML not only enables definition of tag catalogs, but also definition of the rules that relate these elements with the others, as well as mechanisms to further qualify the elements through "attributes". SGML is therefore a metalanguage rather than a language, allowing to create as many description languages as necessary. Each of them is formally described in a "document type definition" (DTD).

SGML has been published in 1986, and has given rise to a number of offsprings and evolutions. One of them, called HyTime, contains a diversity of features that complement SGML and open access to the world of hypermedia/ time-based documents.

SGML is originally based on a publishing model, where collections of documents are assembled into one large document. Each object in an SGML document can be given an identifier, that is unique for one document. In other words, the scope of the SGML unique identifier is the SGML document. Addressing an object means in SGML referencing its unique identifier.

This model has worked well until the question came to relate various documents together. SGML does not have any mean to express, in a standard way, the address of an element belonging to an external document. By working on an addressing model, the creators of HyTime designed a very rich and universally applicable set of address mechanisms. Addressing by identifiers is only one specific case of addressing, among a variety of others.

HyTime location addressing aims at replicating the universality of the bibliographic model, enabling referencing to anything, anywhere, at any time. This model is based on the concept of "Integrated Open Hypermedia". A reference such as "see figure 5 page 25 in R. Dondeg, The Economics of Publishing, Pennywise Press, London, 1902", conforms to this model. It is integrated, because it is possible to address an object which exists, independently of the fact it was planned to be used as an anchor. It is open, because the reference can be found in a library, independently of the shelf number on which it is stored. It is hypermedia, because this reference is a hypertext link connecting a portion of text to a graphic, and therefore can be considered as both hypertext and multimedia, i.e. hypermedia.

HyTime has standardized the notion of independent link. An independent link is an element that relates two or more anchors by declaring the connection externally. It contrasts with traditional kinds of links, qualified in HyTime as "contextual links", of which cross-references or links on the Web are the most frequent examples. Contextual links have an origin and a target. Online, they are most often represented by a hot spot, marked generally by a blue underline text on the Web, which implicitely say: "Click on me, and I'll drive you somewhere else." In the printed world, cross-references (for example, "see also p. 89") are the equivalent of contextual links.

Over the identifier reference in SGML, HyTime's contextual links have the advantage that the target need not be the identifier of a unique anchor, but indirect addressing offers the possibility to insert a location address element in between. This element may point to elements in external documents as well as aggregates of pointers to a number of targets.

Contextual links are more powerful than simple references (unique identifiers in SGML, hypertext references in HTML), but they need to be present in the source document. Therefore, if one of the targets of the link changes (for example, is removed), the link is not valid any more until a corresponding change is made in the source document as well. Therefore, contextual links are easy to create, but difficult to maintain in a context of documents that change often.

Independent links have two advantages over contextual links: they are external both to the source and to the target, and they express the semantics of the relation. In fact, the notion of source and target becomes a specific case for independent links. Traversal is, in most cases, bidirectional, i.e. it is possible to go from the source to the target, as well as from the target to the source. Moreover, because links involve more than two anchors, the very notion of source and target is losing its meaning. Basically, a link aggregates objects that have a certain kind of relationship with each other. The way they are traversed can be considered, in some cases, as a formatting artefact.

An independent link expresses the semantics of various anchors, grouped by the roles they play in the link. Therefore, a link expressed a type of relationship, where every group playing a role is clearly identified. For example, a family can be described as a link with HyTime semantics. A typical western family is composed of persons, one of them being the father, another one the mother, and several possible being sons and daughters. The expression of the semantics of the relationship is differentiated from the location of the elements composing it. Even if children leave their parents' house, they are still members of the family. This is the same when an anchor changes location, it still remains an anchor.

The difference between locating and addressing is the key feature of the design of HyTime that is used in the Topic Map model. Location addresses are considered transparent, or invisible, or irrelevant, as far as the linking semantics is concerned. This separation opens the possibilities of very sophisticated linking/addressing implementations. HyTime's addressing facilities make possible locating practically anything, i.e. an information base stored in any format, including documents, as well as databases, spreadsheets, graphics, videos, or portions thereof; it is possible to address any portion of any document, even if it is not structured.

Also, because independent links are "out-of-line", they can be stored and managed within a link database, offering powerful possibilities for managing the network of information between lots of various sources. Mastering the structure of links gives much

control over information, especially if links contain knowledge, expressed as roles played by various anchors.

Links in HTML belong to the category of contextual links: "Click and go". They are used extensively on the World Wide Web, and have resulted in the most successful information network that ever existed. Everyone is able to connect to everyone else web site. The most popular web sites are those from which it is possible to find other information. Basically, they contain lists of links, built with automatic search engines or by hand.

XML is a subset of SGML created under the auspices of the World Wide Consortium. Version 1.0 has been out in February 1998. XML is the core specification and other recommendations are using it, e.g. the Extended Linking Language, or XLink.

The XML link specification contains two types of links: simple links, which can be considered the same family as HyTime contextual links, and extended links, which contain the basic features of HyTime contextual links. Extended links are "out-of-line", and they are well suited to the representation of Topic maps.

- ISO 13250 is expressed in SGML.
- ISO 13250 is an application of HyTime (ISO 10744)
- It uses architectural forms It uses the hyperlink module.
- Addressing facilities all included.
- ISO 13250 can be an application of XLL (Extensible Linking Language)
- X-Pointers can be used for addressing.
- Extended Links can be used to describe the links.

XML and XLL: Simple versus Extended Links

Simple links

- Easy to create, easy to understand
- Widely used on the Web: HTML links (A HREF pointing to A NAME)
- Widely used in SGML: ID-IDREF mechanism.
- Very difficult to maintain when documents evolve.

Extended links

- Links are defined outside the locations they point to.
- Multiple anchors (aggregates).
- Multiple semantics for anchor roles.
- Make possible the maintenance of links in a separate database.
- Fit to the Topic Map Model.

Management of Information Objects

A topic map is an overlay superimposed on a series of documents or information objects. HyTime has a feature allowing management of a set of information objects, called the "Bounded Object Set" or bos.

Addressing and Linking

- HyTime Location Addressing Module: addressing anything, anywhere, at any time.
- Addressing can apply to structured documents (SGML/XML)
- Addressing can apply to structured databases (queries)
- Addressing can apply to unstructured information flows (e.g. Ascii)

- Addressing can apply to formatted information (e.g. Postscript, PDF)
- Addressing can apply to non-textual information
- Resolving the addressing is an application issue.

Addressing

The ambition of a universally applicable addressing schema for online information is to replicate the classical bibliographic model for referencing, that lets us give a well-understood, unique, way to locate a document, for example a book: the bibliographic reference. It generally contains the name of the author, the title of the book, the publisher, the date of publication. If we are quoting an extract of this book, we can add the page number. This mechanism, although quite old, is very efficient, because it relies on the concept of publishing, i.e. to give a certified, legally acceptable, stamp to a work as a book. When a book is available in a library, it can be searched for, without any knowledge of the system of classification locally in use. The work of the librarians who are creating and maintaining catalogs is to give a local address, for example a shelf number, to a book identifiable by a universally valid bibliographical reference. No matter in which part of the world we are in, since we are in a library and we know the reference of a book, there are ways to find it.

HyTime's location addressing facilities aim at replicating this facility for online information. There, the situation looks more like a mess. Electronic documents have no way to be uniquely referenced, for example by a title and an author. It's more important to know where they are, on which server, on which hard disk, what format they are in, and once a specific information is found, it's not guaranteed that it will be possible to read it, because the format in which it is stored might have been created with a software that is not available any more, or has no equivalent on the machine used to read it. Or it can be stored on a diskette for which there are no more disk drives. This is a major challenge, and we are used to consider online information as volatile and temporary, but it's not any more. It is very likely that our grandchildren will not be able to access most of the information we are producing today. One of the answers to this challenge is to store documents in common, internationally agreed upon, formats, such as SGML or XML. But still, this does not answer the question of how to actually retrieve one specific documents. To answer to this question, HyTime contains a very powerful set of addressing facilities, which can be divided into three categories: addressing by name, addressing by property, addressing by coordinate.

Addressing by name implies that the information object which is searched has a name. This name must be a unique identifier, that qualifies either the document itself, or any amount of information within the document.

Addressing by coordinate means to count for occurrences of identical objects until the one that we are looking for is found. For example, third paragraph of the fourth chapter of a given book. Actually, this is an extension of the page numbering facility, except that it can be applied to something else than pages. Pages are only meaningful in electronic world when the delivery software produces pages that emulate the printed pages. PDF, for example, is a format that produces pages that resemble very much to pages of printed material.

Addressing by property is the same as querying an information. For example, if one is looking for all documents in which one specific word appears, such as "elephant", this is an address by property, because it is like saying: I want to see all documents which share the property to have the word elephant at least one time inside them.

Addresses are represented by special elements, also called pointers, which indicate a location where an information can be found. A pointer can be a name, a coordinate, or it can express a property. It is possible to combine different address elements, by making one of them point to another one. The address is said to be resolved when it eventually returns an information object that is not itself an address. There are no limits to the number of address elements that can be interposed between the initial pointer and the information object.

Address elements may contain a pointer to one object, as well as to several objects. When several objects are grouped into an address, the group is called an "aggregate".

When an address element, instead of pointing to an information object, points to another address element, it is called "Indirect addressing". Indirect addressing seems complex at first sight, but can be very useful as soon as there is more than one object that

should be referenced into an address.

Addressing can be done for any existing format. However, the fact that a format is or not addressable depends on the application. In general, applications address a list of formats, which is closed and well-defined. A format which is not in the list is not addressable by this application.

SGML/XML documents are not the only kinds of information objects that can be addressed. Databases also can be addressed by means of queries. Unstructured information also can be addressed, by any kind of mechanism.

Addressing information is one of the most painful things to do, when creating a complex information base. For example, creating an index by hand requires patient handling of page numbers, that must correspond to the places where the indexed term occurs. Each time there is a modification that obliges to repaginate the document, the index has to be re-created. Doing this by hand is very time-consuming, and is an awful task. This is the way it was done, though, until word processors software start to include automatic processing for indexes. But this method works well for documents that resemble to books. It's not appropriate for bigger sizes.

Linking

Linking information is traditionally done using hypertext techniques. In its most basic form, a hypertext is a text containing hotspots that enable traversal to another information object, located either in the same document, or in another one. There is a notion of "origin" and "target" in these links, limited to two anchors. Anchor is a generic term used to qualify the information object which serves as one end of the link.

In this view of hypertext, the link is more or less a pointer to another address. It makes it straightforward to use. Usual user interfaces for creating link enable opening of simultaneous windows and after triggering the link facility, the user clicks in the target in order to create the link. This interface is not as user-friendly as it seems, because it only offers the feature of creating links by hand, one by one. It is not a method that can be used efficiently as soon as the number of documents becomes important. And the worst is that when something changes in a document that is a target of a link, the information contained in the origin of the link becomes erroneous, or obsolete. The link becomes "broken". It points to nothing. This error is frequent on the Web and has become famous as "Error 404".

There is no easy way to get around this problem when using this kind of link, called "contextual links". Because origin and target are disconnected, the only way to check for link integrity is to manage origins and targets together, and to treat the link as a separate object on its own. This is the origin of what is called in HyTime the "independent link", and in XML an "extended link".

An independent link handles separately the set of addressses that are involved in the relationship, and the semantics of the relationship. The previous kind of link, contextual, has an implicit semantic for each of its participating constructs, i.e. "origin" and "target". But this is only one among an infinity of possible relational semantics between anchors.

In the independent link model, there can be multiple anchors for a link, and not only two. Each anchor can be itself a single object, or a set of objects. What is called anchor here is the group of information objects that is involved in a given relationship and plays a specific role in that relationship. For example, a family link (simplified) can be described with 3 anchors: mother, father, children. The anchor called "children" may include zero or more persons; the anchor called "mother" and "father" usually include only one instance, although there are [many] cases where the situation is not as clear. For example, in a family. a stepmother may play the role of the mother and the actual mother may have created another family, but is still considered to be the mother in this one. It is a modelling decision under the control of the link model designer to decide whether the anchor role "mother" is allowed to refer to more than one person.

The independent link paradigm therefore enables the description of links independently of any of its anchors, and specifies, for each anchor, the addresses of the objects participating in this anchor, and the specific role that this anchor plays in the overall relationship described by the link element.

The addressses can be quite sophisticated, and use all addressing facilities mentioned above. The semantics of the anchors are differentiated by being expressed as anchor roles.

The fixed versus unfixed anchor role issue

An independent link has two major characteristics: its name, and the list of anchor roles it contains. There are two ways to interrelate a link type with the list of is anchor roles. The first way is to claim that a link element type entirely defines the list of all possible anchor roles. The second way is to make the list of possible anchor roles independent of the link element type.

The first case describes constrained environments, where all possible anchor roles have been predefined for each link element type, and therefore applications must prevent users to invent their own anchor roles, in supplement to what already exists.

The second case describes open environments, where each user is able to add new anchor roles whenever necessary, without breaking the link element type declaration.

In HyTime, there are three independent link architectural forms, called *hylink* ("hyperlink"), *ilink* ("independent link"), and *varlink* ("variable link"). Hylink and ilink are constrained: the list of anchor roles must be defined each time a link element is declared, in the document type definition. Varlink is not constrained. It may contain as many anchor roles as desired, without requiring any changes to the link element type declaration.

In XML, only one form for independent link exists. It is called "extended link". It is not constrained, and has characteristics that are very similar to those of varlink in HyTime.

Unconstrained link architectural forms can be constrained.

The link forms that derive from constrained architectural forms (hylink and ilink) can not be unconstrained. However, the reverse is not true. Unconstrained architectural forms can be constrained. This possibility is due to the defaulting mechanism defined in varlink. The role played by an anchor is either declared as an attribute of the anchspec element form, or, if not declared, the generic identifier is used instead. Constraints applying a generic identifier can be defined in the DTD.

Non-constrained varlink

```
<!element family
           -    O (anchspec)+ >
<!attlist family HyTime  NAME   #FIXED  varlink >
<!element anchspec - O (%locs;)+ >
<!attlist anchspec
          HyTime    NAME  #FIXED  anchspec
          anchrole  NAME  #REQUIRED
>
```

Note that anchrole attribute is required here, in order to allow for maximal flexibility. The name of the element is fixed, but users can create as many anchor roles as they want.

Example of instance

```
<family id="simpson">
  <anchspec role=father>idforJoe</father>
  <anchspec role=mother>idforSusan</mother>
  <anchspec role=children>idforAnn idforDavid</children>
</family>
```

Constrained varlink

```
<!element family - O (father?, mother?, children?) >
<!attlist family
  HyTime  NAME  #FIXED  varlink
>
<!element father - O (%locs;) >
<!attlist father
  HyTime   NAME  #FIXED anchspec
  anchrole NAME  #FIXED "father"
>
<!element mother - O (%locs;) >
<!attlist mother
  HyTime   NAME  #FIXED anchspec
  anchrole NAME  #FIXED "mother"
>
<!element children - O (%locs;) >
<!attlist children
  HyTime   NAME  #FIXED anchspec
  anchrole NAME  #FIXED "children"
>
```

Note that the anchrole attribute has a fixed value, which is the same of the generic identifier.

Example of instance

```
<family>
 <father>idforJoe</father>
 <mother>idforSusan</mother>
 <children>idforFred</children>
</family>
```

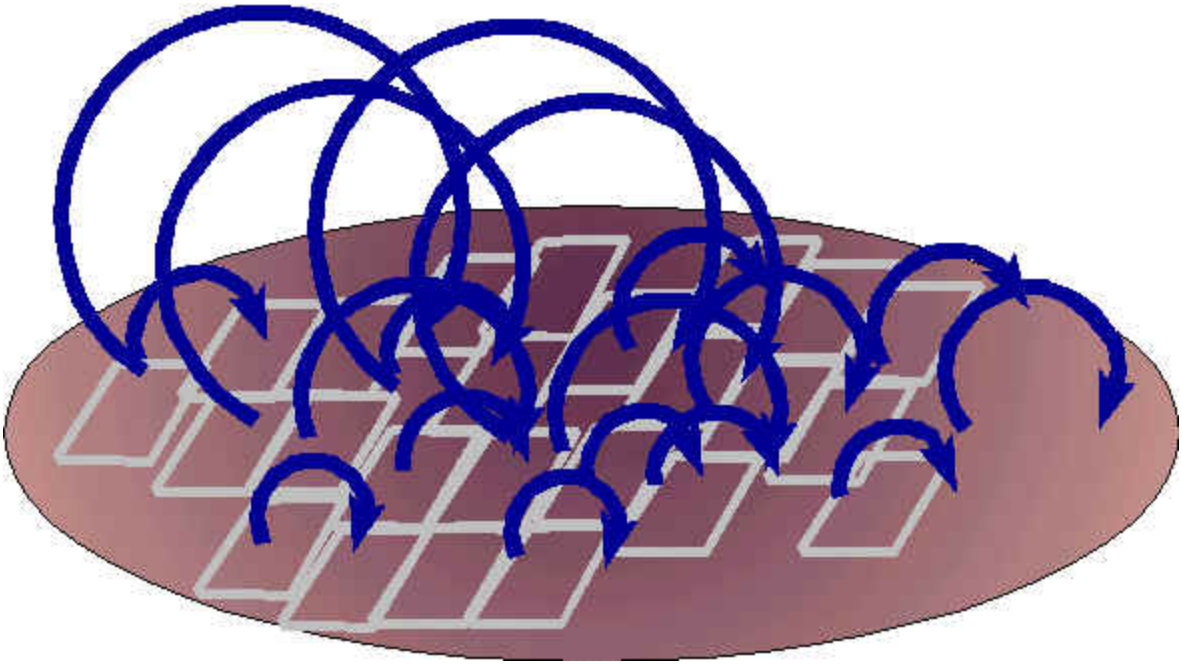
Syntactical differences versus equivalent representations From the point of view of the HyTime grove, the two representations

and

are equivalent, if the element father has been declared as deriving from the "anchspec" architectural form. Therefore, whatever syntax has been actually used is indifferent at the time of processing.

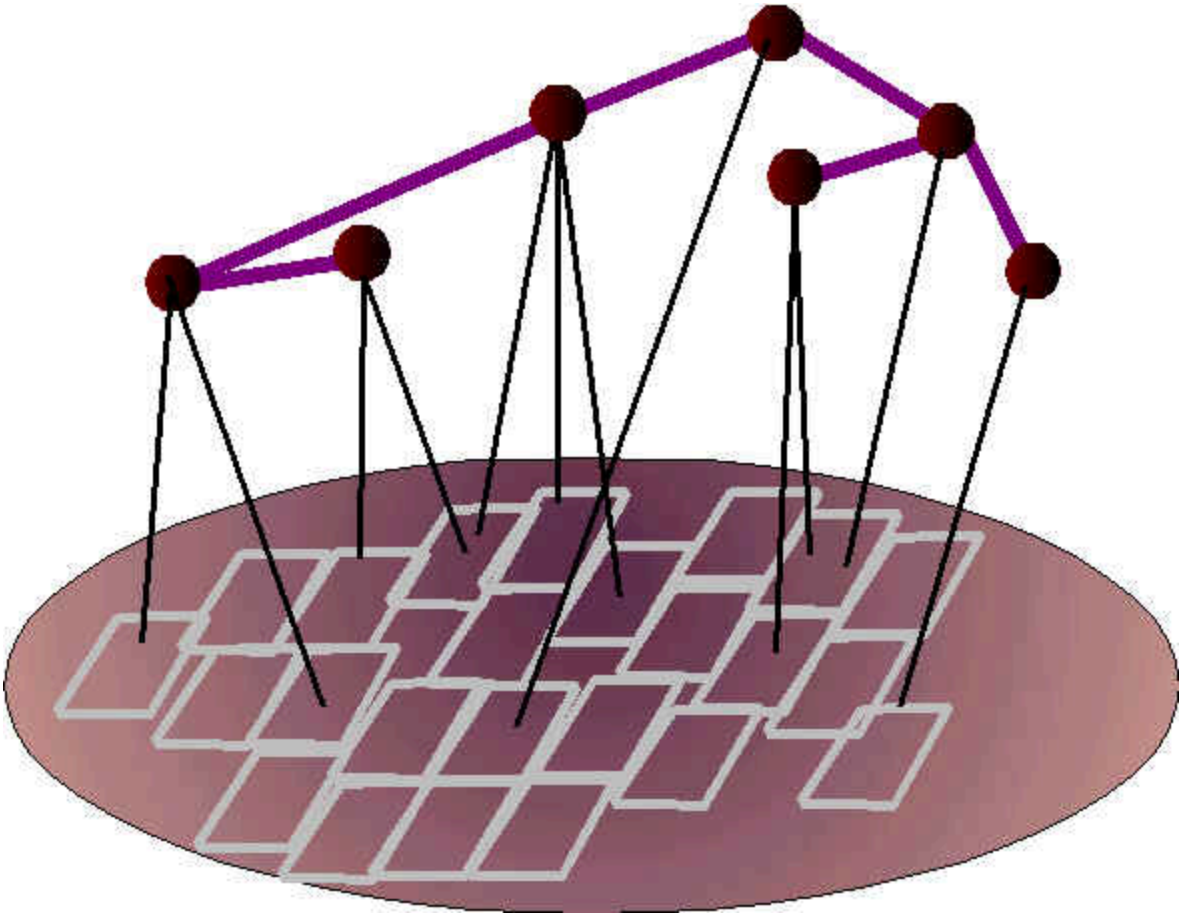
Contextual versus Independent Links

Contextual links have one of their link ends being the link itself. This is the origin of the link. Contextual links are widely used. They are called "cross-references" in the books. Contextual links are simply called "links on the Web.



An independent link database

- Links are created independently of the documents or information sources to which they point.



- Therefore, Topic maps can be used to describe structured or non-structured information repositories.
- Topic Maps can be used to connect heterogeneous information sources.

The Topic Map Model

- Topic
- Association
- Facet

The Topic map model describes how to use unobstrusive overlays to express navigation and filtering strategies of information objects. The model defines templates that topic map designers can use to describe their navigation model and filtering strategy. The three architectural forms: topic, association, facet, correspond to a different aspect, well identified, of the description of information.

The Topic Map model is made of three basic constructs, which are links, plus a mechanism to give alternative names.

The three types of links are called:

- Topic link
- Association link
- Facet link

A **topic link** is used to collect all information objects about a given topic, including its name(s), and several types of specialized occurrences in which it can be found. Occurrences are grouped under user-defined occurrence types. "Definition", "description", "mention", are among possible occurrence types.

An **association link** is a link relating two or more topics together. This type of link can be used to organize topics in hierarchical order, or more generally to create a knowledge base defined with the topics as atoms.

A **facet link** is a link that qualifies information objects with properties that will be used for selecting or filtering, allowing partial views of topic maps to be created. This link can point not only to information objects, but also to topics and associations, if they need to be qualified with a selection criterion.

Generic link architecture

TNM constructs are expressed as links.

- **TNM:** Three possible value: Topic, Association, and Facet
- **Type:** Group instances belonging to a given class.
- **Anchor role:** Semantic of the role played by a (possibly) aggregate anchor.
- **Anchor address: Any address element or mechanism possible. Left to the application.**

A link basically is made of anchors. Each anchor is an aggregate of addresses to which the links point and which are grouped following the role they play in the relationship described by the link.

The TNM architecture uses three kinds of links. They all inherit from the same hyperlink architectural form. The three possible values are "Topic", "Association" and "Facet". Each of those links is expressed using the varlink syntax.

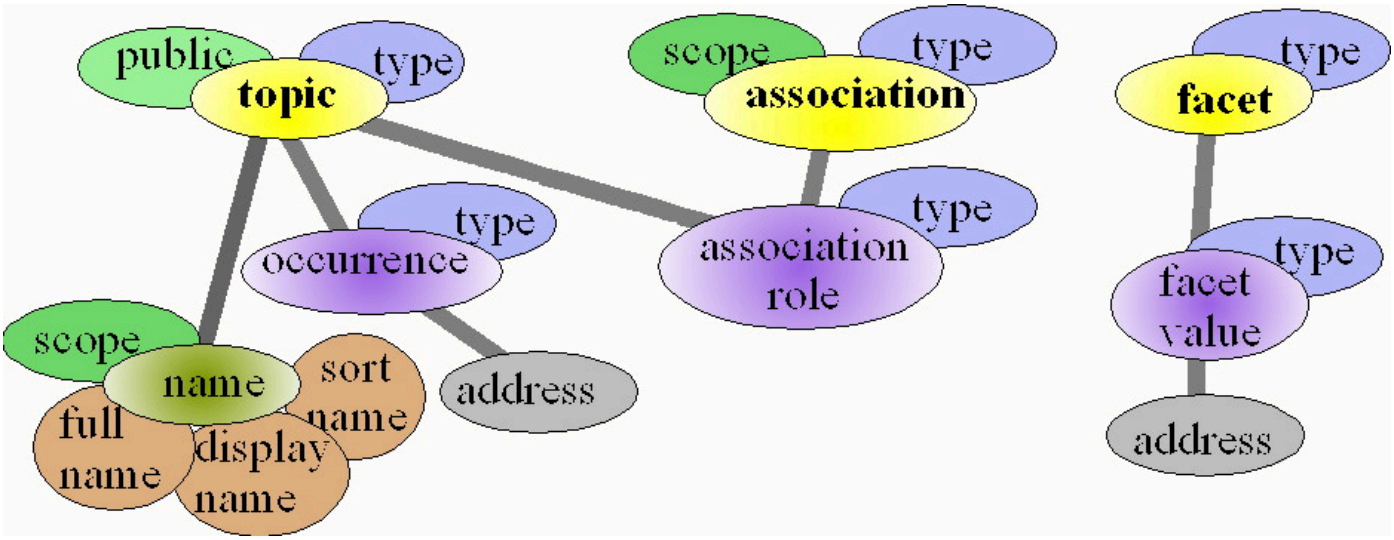
The "type" attribute is used to specialize topics. For example, "painter" can be used as the category for all topics about painters, "language" for the category "facet" link containing various languages of the world, and "containment" can be used to categorize an association.

The TNM Model at a glance

```
                  Contains             Type      Other
LINKS ---------------------------------------------------
topic             name, occurrence     type      public
association       association role     type      scope
facet             facet value          type

ANCHOR SPECIFICATIONS ------------------------------------
occurrence        address              type
association role  address              type
facet value       address              type

SPECIFICS ------------------------------------------------
name              full name,                     scope
                  display name,
                  sort key
```

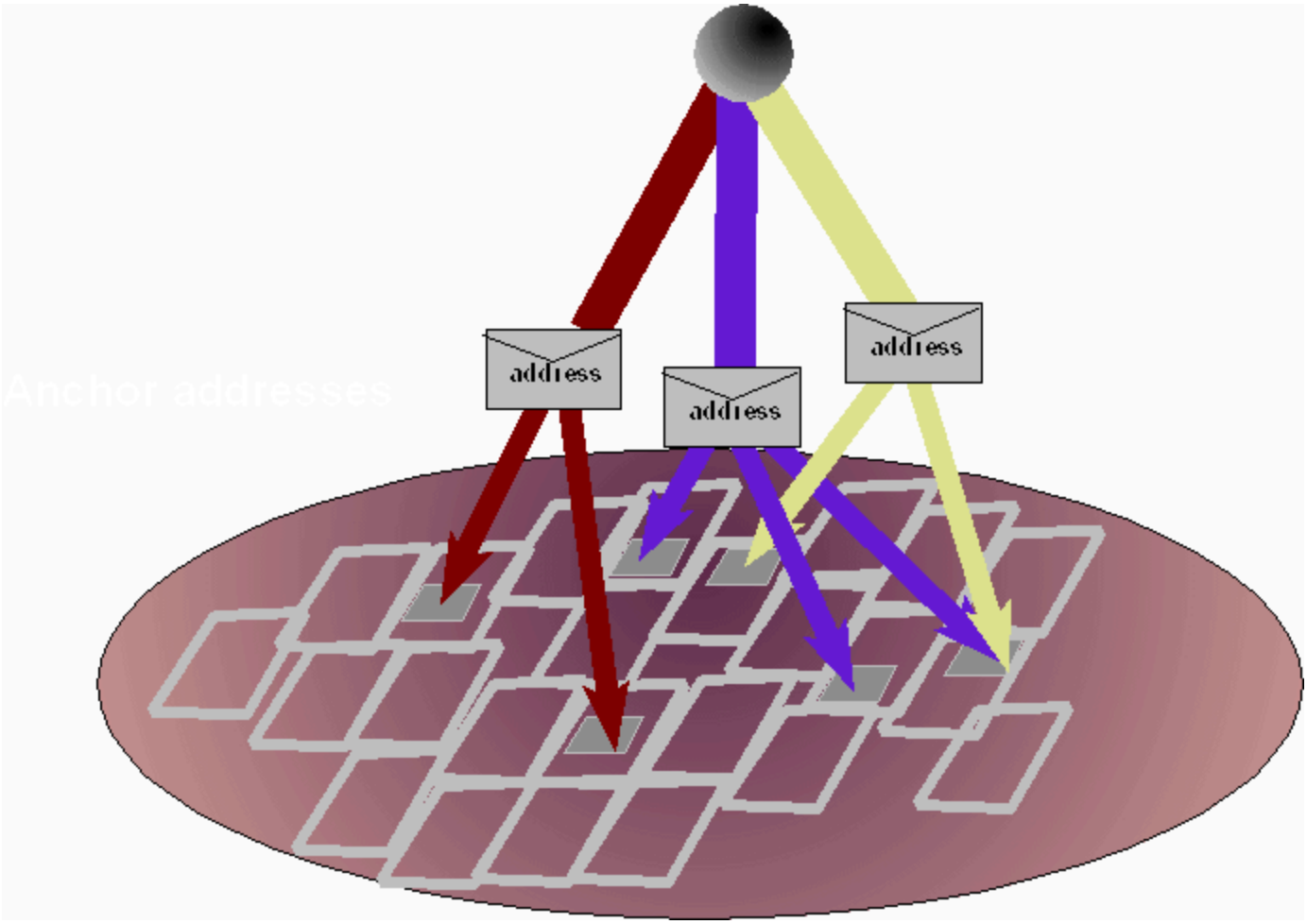


!(graphics/tmmodel.jpg "Topic Maps Model")

Types and *scope* scopes can be considered as topics. Therefore, a topic map can be used on itself. Types are used to point to topics, which make the object itself like a topic. In this perspective, the only things which are not topics are really addresses of information objects *information object* . This diagram allows to understand the notion of "integral topic map". Everything is potentially representable as topics within a topic map, including the topic map constructs themselves.

From the point of view of the link syntax, the topic, association and facet elements are the *link element types* . Occurrences, association roles and facet values are the *anchor roles* for each of these link types.

Topic

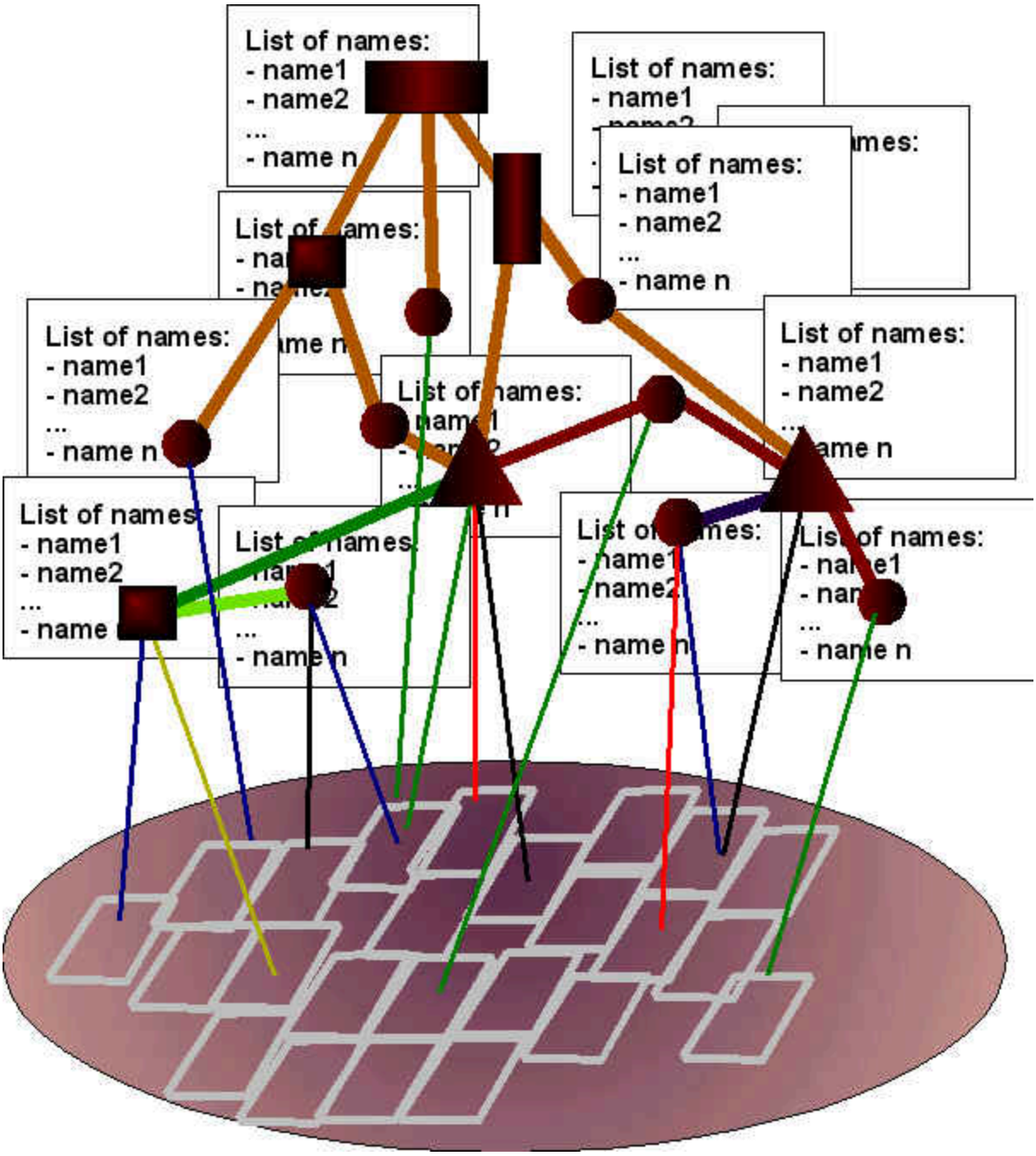


- A topic is a link that aggregates all locations about a given subject.
- Topics can be typed. Any number of topic types can be created.
- Occurrences of a topic can be distinguished by the role they play.
- Any number of occurrence types can be created for every topic type.
- A topic may have zero, one, or several titles.
- A topic without a title is like a cross-reference.
- Filters (see below) can be used to select one among several titles (e.g. in a given language).
- A topic may, or may not, have a definition. Definition is considered one specific anchor role.
- The Topic form can be used to create and manage indexes, glossaries, and cross-references.

Topic is a word whose greek origin (*topos*) means both a subject and a location. In the Topic map model, topics collect all pieces of information which happen to be on a given subject. This set of locations is what makes sense, what creates a meaning. A topic is therefore a complex construct, because many things can be said about a given subject. Once a topic has been built, the topic structure is taken as a whole, and all its components participate in a black box, that is considered globally when operations are performed on the topic. Among this variety of things that characterize a topics, some are defined by the Topic Map architecture, and some are left to individual topic map designers. Topics have two sets of characteristics: names and occurrences.

Topic Names

The name of a topic is one of the things that participate to build a topic. The name of a topic is not intrinsically attached to the topic, but instead is a separate object, connected to the topic, and where it is clear that the object is used as the name for the topic. The reason why this indirection exists is because it is too limiting to consider that a topic has a name. Some topics might have no names, and some topics might have several names. Furthermore, any topic name can have several alternative forms.



Let's consider first a topic that has more than one name. There might be several reasons for this to happen. One case is to provide the possibility for topic map users to access to a topic by indifferently using one among several names, which are equivalent. As an example of this situation, consider a topic whose name is "art museum". Another possible name would be "museum of art". We suppose that the topic map designer considers these two expressions as rigorously equivalent, and do not have any preference on whether to use one expression rather than the other. In a printed index, this situation would result in two possible cases: the two entries "museum of art" and "art museum" be followed by the same set of page numbers, or one refering to the other, such as in "museum of art: see art museum". Doing the same in an electronically browsable topic map would result in a waste of time. It is more efficient to directly go to the common set of occurrences, whatever names have been chosen as an access point to the topic. The fact that the two names share every other characteristic for the attached topic, results in the fact that only one topic exists in a topic map. Therefore, each time a modification will occur, for example if a new occurrence shows up for the topic, it will automatically apply to all the names used to access the topic.

Another important case where a topic has more than one name is when filtering is done prior to accessing the topic on selection criteria. Filtering will result in displaying only one name in a set of names available for a given topic. An example of such a case is a multlingual topic map. Each name will be seen by applying a given filter and the topic will then be only a partial view of the

original one. It's like if the topic would have characteristics that have all the colors of a rainbow, and the ones that would appear after a filter is applied are only those which have a given color. In the previous example, two names could be added to the same topic: "musée d'art" and "musée des beaux-arts", which are two equivalent names to express the same topic name, but are only to be viewed when the filter "French version" is applied.

Topic names	museum of art			
		art museum		
		musée d'art		
			musée des beaux arts	

Topic names:

Topic names

- museum of art
- art museum
- musée d'art
- musée des beaux arts

The topic here has four names. They are all equivalent as far as the topic itself is concerned, and they all give access to the same topic. If filtering mechanisms exist in the topic map, then the names "museum of art" and "art museum" can both be declared to be relevant to the "English" facet, while "musée d'art" and "musée des beaux arts" can be declared to be relevant to the French facet.

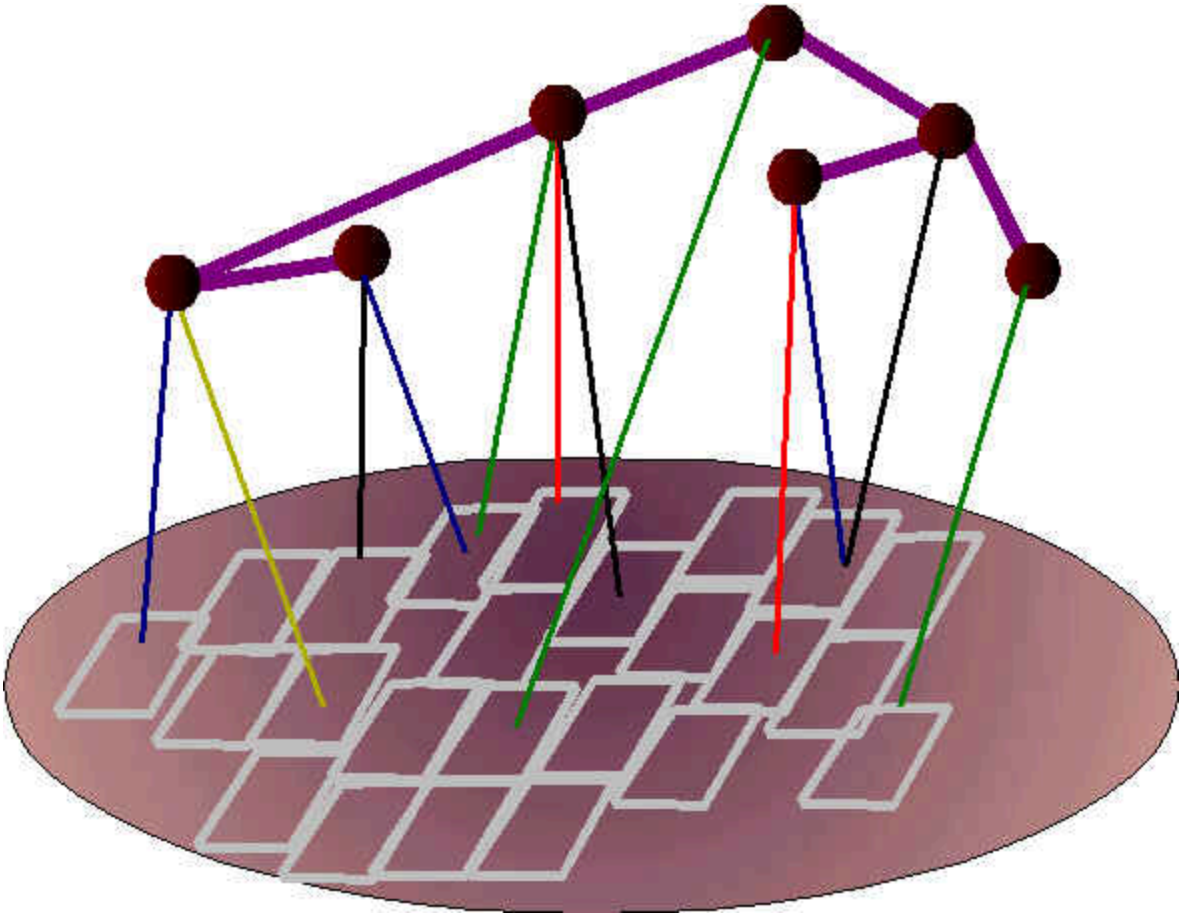
Topic without names

Topics are not only accessed by their names. They can also be accessed by entering one of their occurrences. For example, it is possible to desing software enabling jumping from one occurrence of a topic to another, regardless how the topic is called. This is a function ordinarily performed by references: either cross-references, or bibliographic references. The difference between a cross-reference and a bibliographic reference is explained in this book (page ...). The previous sentence itself expresses a cross-reference. It basically says: to know more about the subject I am talking about here, i.e. "the difference between a cross-reference and a bibliographic reference", can be constructed as a topic, which has two occurrences, one being the source (here) and the other being the target. The idea here is: each time there is a connection between two objects located in different locations, the connection is made because there exists some kind of relationship between these objects. If the relationships expresses the fact that these objects share the property to be about a given subject, then the relationship can be considered to be simply the fact that these two objects are in fact occurrences of the same topic. Usually, the topic name is not explicitely present, it is deductible from the context. The deduction is obvious for a human reader, but very difficult to derive for a computer. Transforming cross-references into occurrences of topics makes then understandable as well for machine processing. It considerably helps for the maintenance of evolving information bases containing many cross-references , a problem which has proved to be quite difficult to solve in a general way.

In summary, topic without names is what can be automatically derived from extracting existing cross-references. They can therefore become part of a valid topic map, since the Topic map architecture accepts topics with no names. They can be gradually "upgraded" by adding names, therefore adding -- or duplicating -- index entries, in order to gain more user control on topic map navigation. But this strategy is not required by the topic map standard. It is only made available by it.

Topic occurrences

Topic occurrences are all information objects that are related because they are all about the subject optionally described by the

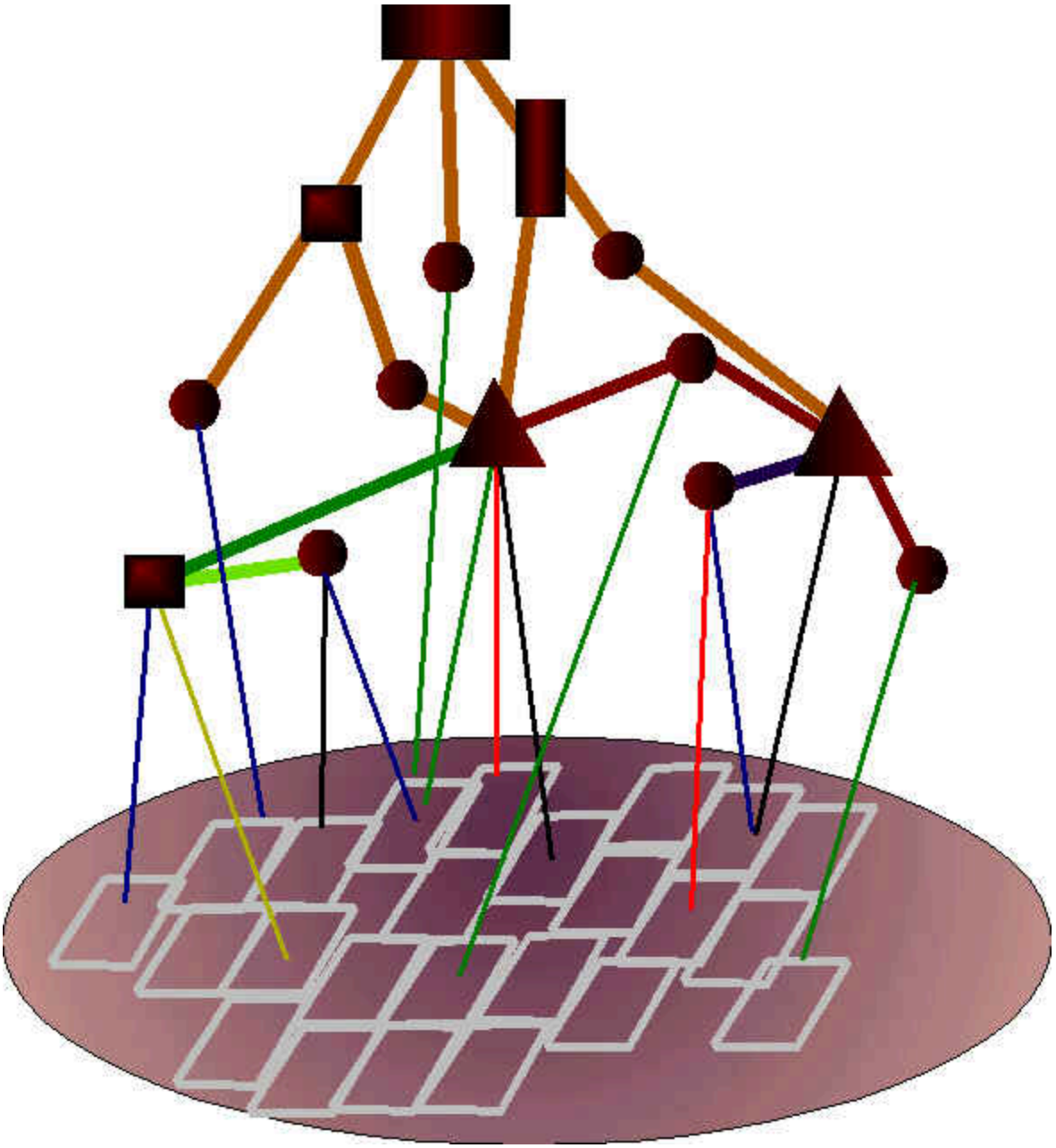


topic names.

The nature of the objects being related is left open to the application. They are expressed in various formats, and topic map applications are able to address a list of formats that are defined. For example, if the topic map application is able to understand structured document formats such as XML, SGML or HTML, then any element that is defined with these formats can be used as an occurrence for a topic. A topic occurrence can extend from an entire document, to a paragraph, or even a single word. The topic map application might as well be able to consider as an occurrence a single word, even if this word is not itself an element. If the topic map application is able to address a database format, then an occurrence can be a record in the database, or even a field in a record. If the topic map application is able to address page-based formats, such as Adobe's Portable Document Format (PDF), then any rectangle drawn on a page, containing either text or graphics can be used as an occurrence for a topic. If the topic map application understands ASCII (or Unicode) text, then any word or group of consecutive words can be considered the occurrence of a topic. And, if the topic map application understands multiple formats, than any mixture of the above can serve as topic occurrences.

The various ways how to declare where topics are located to let the application search for topics and mark their occurrences is left to the applications. For example, an application may be able to find the paragraphs in which a given word occurs and mark them as occurrences for the given topic. More sophisticated conditions can be given to exploit the existing data. The nature of such kinds of processing is application-dependent. The more source information objects are structured, the easiest it is to automatically build a topic map to retrieve this information. At the other end of the spectrum, automatic indexing and search engines can be used to locate occurrences of topics starting from scratch, and the quality and efficiency of the matches found depend on the reliability of the software application used to create them.

Topic Types



Occurrence types

The topic map architecture enables designers to say more than locating occurrences of topics. It is also possible to subdivide occurrences by classifying them into common roles they play. It can be useful to distinguish between occurrences which are descriptions, definitions, portraits, biographies, warnings, notes, graphics, etc. These occurrence types can be defined, if this is meaningful, as depending on a given topic type. For example, "portrait" and "biography" will only apply to the topic type "person", while "description" will only apply to the topic type "product". The definition of occurrence type is left entirely open to the topic map designer.

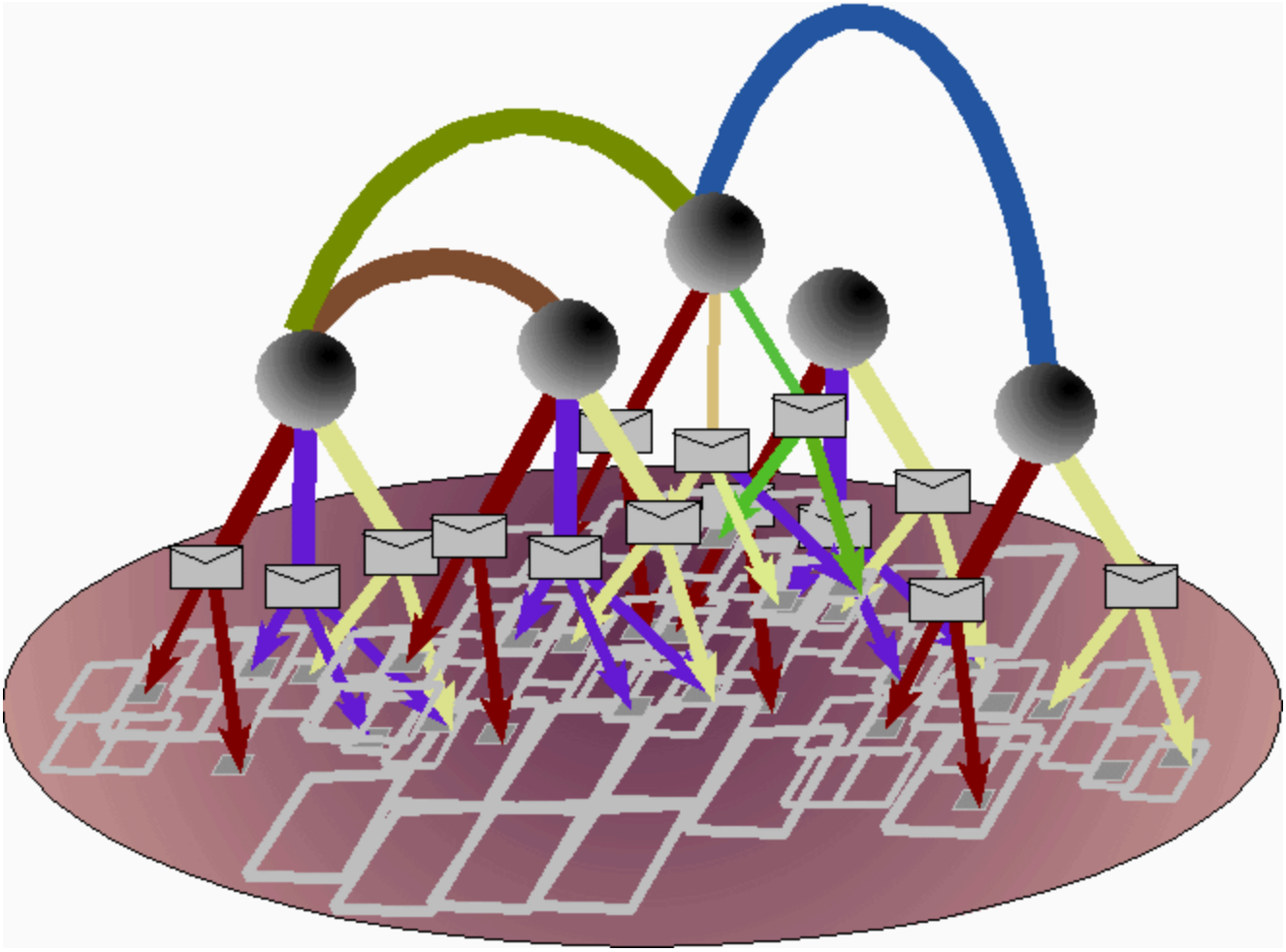
Topic occurrence types can be either automatically derived from already existing structures (document markup or database fields), or added manually, or in a partially automated process, to unstructured information. It can be used to "structure afterwards" information sources that were not previously structured, or to merge between incompatible structures by applying a common structure above the existing ones, by saying for example that the existing elements "warning" and "caution" coming from different sources both correspond to the same same occurrence type unified under the name "warning" for example.

Topic occurrence typing can be used to emulate what is used in classical printed indexes, i.e. the fact that main occurrence references are sometimes printed in bold. With this model, instead of being limited to typographical appearance, what appears in the occurrence type is the exact semantics that the topic map designer will choose to describe.

Definitions

Definitions are specific types of occurrences that can prove useful in some topic maps. Considering definitions as specific occurrences rather than constructs on their own helps to point to already existing definitions, if any, instead of having to create a separate set of definitions, stored in a distinct glossary. A topic might have zero or several definitions. Several definitions can be made available simultaneously, or topic map designers can decide to limit the number of definitions to one, and use filters to display one rather than the other, depending on the selection criterion applied. For example, it is possible to mix in an information set, definitions that are only applicable in a list of given facets: one definition would for the beginner's level, another one for the expert level, and a third one for the intermediate level, if a topic map is used as a source for navigating training material.

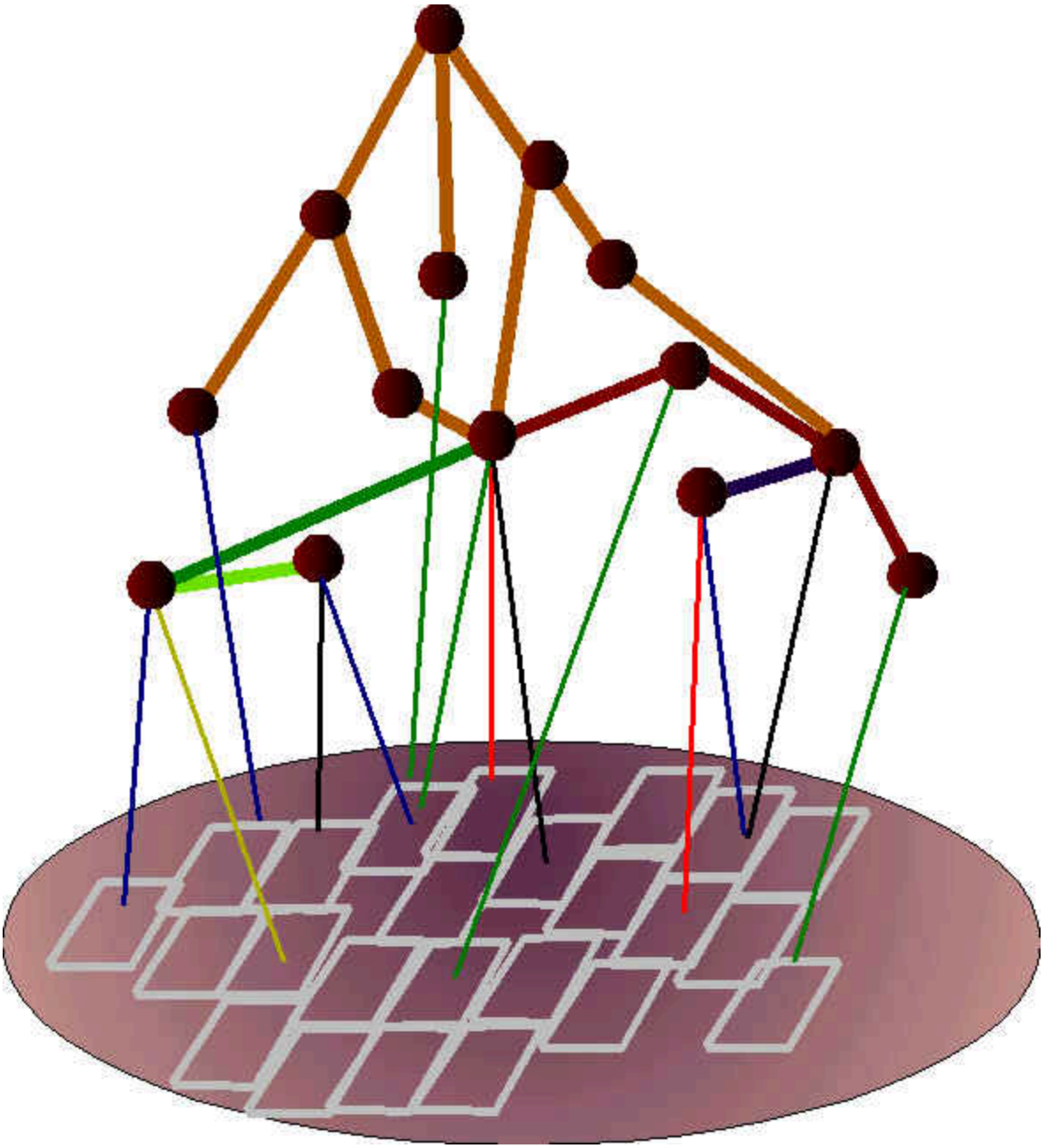
Associations



- Topics can be related together through topic relationships.
- Topic relationships only apply to topics.
- Topic relationships comprise a knowledge base, organized like a relational database.
- Topic relationships can be used to create and manage virtual tables of contents, catalogs and thesauri.

Topics can be related together through topic relationships. Once topics have been built by collecting information objects sharing the semantics of being about a given subject, they can be assembled and related together. The set of interconnected topics is

collectively called a topic map.



Topics can be grouped because they belong to the same group, or the same type. For example, all topoics about given persons can be grouped under the topic "person", through a relationship expressing the fact that "Pablo Picasso" is a "painter". The relationship "is a" is made between the topic "Pablo Picasso" and the topic "painter". Note that when I refer to a topic called "Pablo Picasso", I am really speaking of a topic whose name contains the string "Pablo Picasso". Likewise, the topic called "painter" is really a shortcut to refer to the topic whose name is "painter". In this example, it's likely that the topic "Pablo Picasso" wil have a number of occurrences while the topic "painter" might not have any other occurrences than its name. This is one example of a case where a topic (painter) only existes because it serves as a container (i.e. the anchor of several "is-a" relationships) for other topics. This generic relationship captures the distinction made in database schemas or SGML/XML markup between types and instances. In a database environment, there would be a database of painter records: each record has a field called "name", and Pablo Picasso is the content of the "name" field of one record.

```
painter
name:Picasso
```

In SGML/XML, the name of the type can be instantiated as a generic identifier, while the content of this element is considered the instance:

```
<painter>Pablo Picasso</painter>
```

Any kind of relationship can be created using topic relationships. The semantics expressed by the relationship is open to the topic map designer. Relationships can be created between topics that are describing various types of things.

This example shows how it is possible to design a set of topic relationships to navigate in a world's touristic atlas. A visitor in New York City will find there is a museum called MoMA, while a visitor in Avignon will discover that this city has inspired the name of a painting by Picasso.

Topic relationships can serve to describe the kind of relationships described with entity/relationship diagrams in a relational database. They can also be used to describe the kinds of relationships that belong to various models of thesauri, whether hierarchical or graph-oriented.

The level of flexibility enabled by the topic map architecture is so large that ti may be useful to add constraints to improve consistency in the topic map design. Note however that these constraints are not defined in the underlying standard and are only useful in the context of applications. These constraints can be of various kinds. For example, in the previous example, the relationship whose semantics is "is a country in" can only connect a country with a continent. The relationship between instances and types is expressed by the supplementary following relationships:

```
Pablo Picasso <-- is a-- > painterLes Demoiselles d'Avignon <-- is a-- > painting
MoMA <-- is a-- > museum
New York City <-- is a-- > city
Avignon <-- is a-- > city
New York State <-- is a-- > state
USA <-- is a-- > country
France <-- is a-- > country
America <-- is a-- > continent
Europe <-- is a-- > continent
```

The expression of constraints therefore is as follows: The relationship "is shown at" only applies in this example between a painting and a museum. The example might be more sophisticated, and this contraint could be extended to various pieces of art that are displayed in a museum, and not only paintings. For example, photographs, sculptures, etc., can also be part of these constraints. Here is the list of constraints that may apply in this examples.

- "is a place to visit" connects "museum" with "city"
- "is a city of" connects "city" with "state" or "country"
- "is a state of" connects "state" with "country"
- "is a country of" connects "country" with "continent"
- "originates from" connects "painting" with "city". But this relationship should probably not be constrained here, because many things, not only cities, can serve as inspiration for paintings.

These sets of constraints are not defined by the Topic Navigation Map Standard itself, because what matters for the standard is the resolved result. The many possible methodologies and checkings that have been created to design and build a meaningful topic map are part of the application. There are so many possible ways to reach a given resolved topic map that the process of making it is not defined.

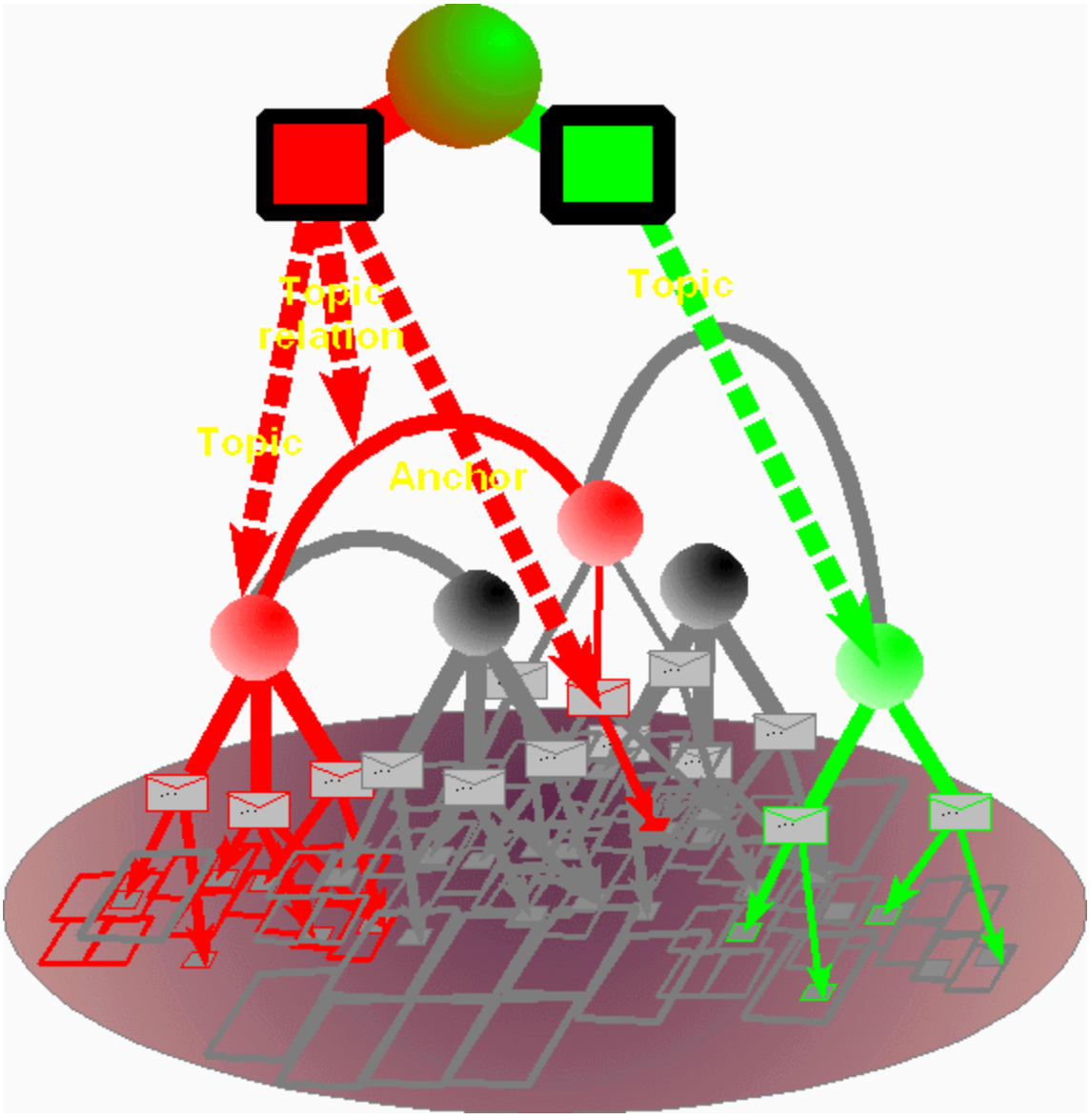
The previous example can also be written with less constraints. Instead of having multiple relationship types, a minimal number could be used to give the following appearance:

```
MoMA is in New York City
New York City is in New York State
New York State is in USA
USA is in America
```

In that case, the navigation model will be equivalent, but the semantics of the relationship will be looser, and it will therefore be more difficult to check whether the information has been correctly entered.

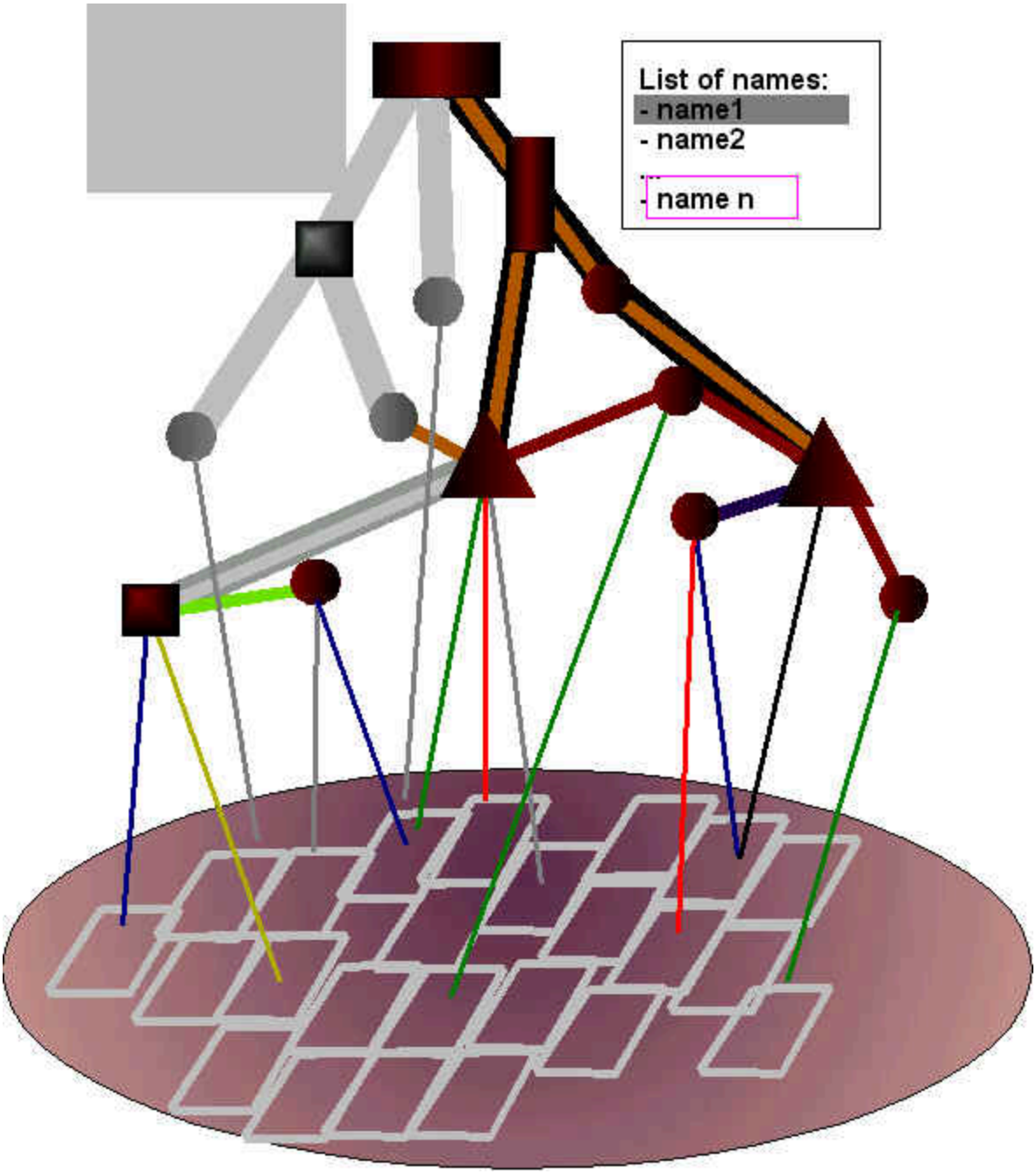
This discussion shows that the more specific relationships are, the easiest it is to check the consistency of the information entered.

Facet



- Facets can be applied to topics, topic relationships or anchors.
- Facets are links that apply to constructs from outside.
- They can be used either to:
  - select partial views of the Topic Map
  - include topics
  - exclude topics

- Examples of facets: language, domain, security, validity, user profile, etc.
- Definition of actual facets is left to the application designer.



When navigating large information repositories, the important point is to access quickly to the relevant information. This can be done by using topics and topic relationships, but it may not be enough if several types of access are simultaneously offered. Therefore it is important to find ways to filter information corresponding to criteria defined by the topic map designers, such as language, user profile, security level, semantic universe, or any combination of the above.

Filtering by language

Topic maps can be rich and complex knowledge bases applying to information objects that are expressed in various languages. The topic map constructs themselves, such as topic names, topic relationship names, occurrence type names, etc., can also be expressed in different languages, enabling speakers of a given language to navigate the map using only their own language. In order to achieve this result, topic maps must provide multiple concurrent views of the same information set, letting only the one which has been filtered to be displayed.

Multilingual topic maps can also be used to only show the occurrences that are in the chosen language. There are many combinations possible for organizing multiple language topic maps. It is a great advantage to be able to have only one structure for organizing topics and use a filtering mechanism to only show what is relevant in one language: the various versions in the different languages are obviously synchronized, because there is really only one version of the topic map.

Filtering by user profile

When a topic map includes information which is made available to identified audiences, the user profile can be used as a selection criterion to only show what is relevant for this chosen audience. This situation can be useful for example when push technologies are applied to send some information to users who conform to a given profile. It can also be used for training material addressed to beginners versus experts. It can be used to distinguish maintenance procedures for technicians from product description for end users.

Filtering by security level

When portions of information can only be seen by a limited audience that is authorized to see it, this is another example of the use of filters to display parts of information repositories. This applies not only to critical information, but also to information that is only of interest to managers versus information which is of interest to the rest of the staff; it also applies to the distinction between information reserved for internal use versus information made available for public use, reflecting the distinction between Intranets and Web sites information contents.

Filtering by semantic universes

Views of topic maps built for describing several domains of knowledge, such as encyclopaedias, can be split into domains, e.g., medecine, art, religion, science, law, etc., in order to facilitate navigation within one limited domain. Filters are also used to provide this facility.

How applications implement filters

A filter can be used by the topic map owners to create a partial view filtered on one, or a combination of, selection criteria. For example, several Cd-Roms can be produced from the same information set, one corresponding to "civil law", one to "business law", one to "international law", one to "fiscal law", etc. Security levels can be applied to prevent unauthorized users to access parts which are not made available to them.

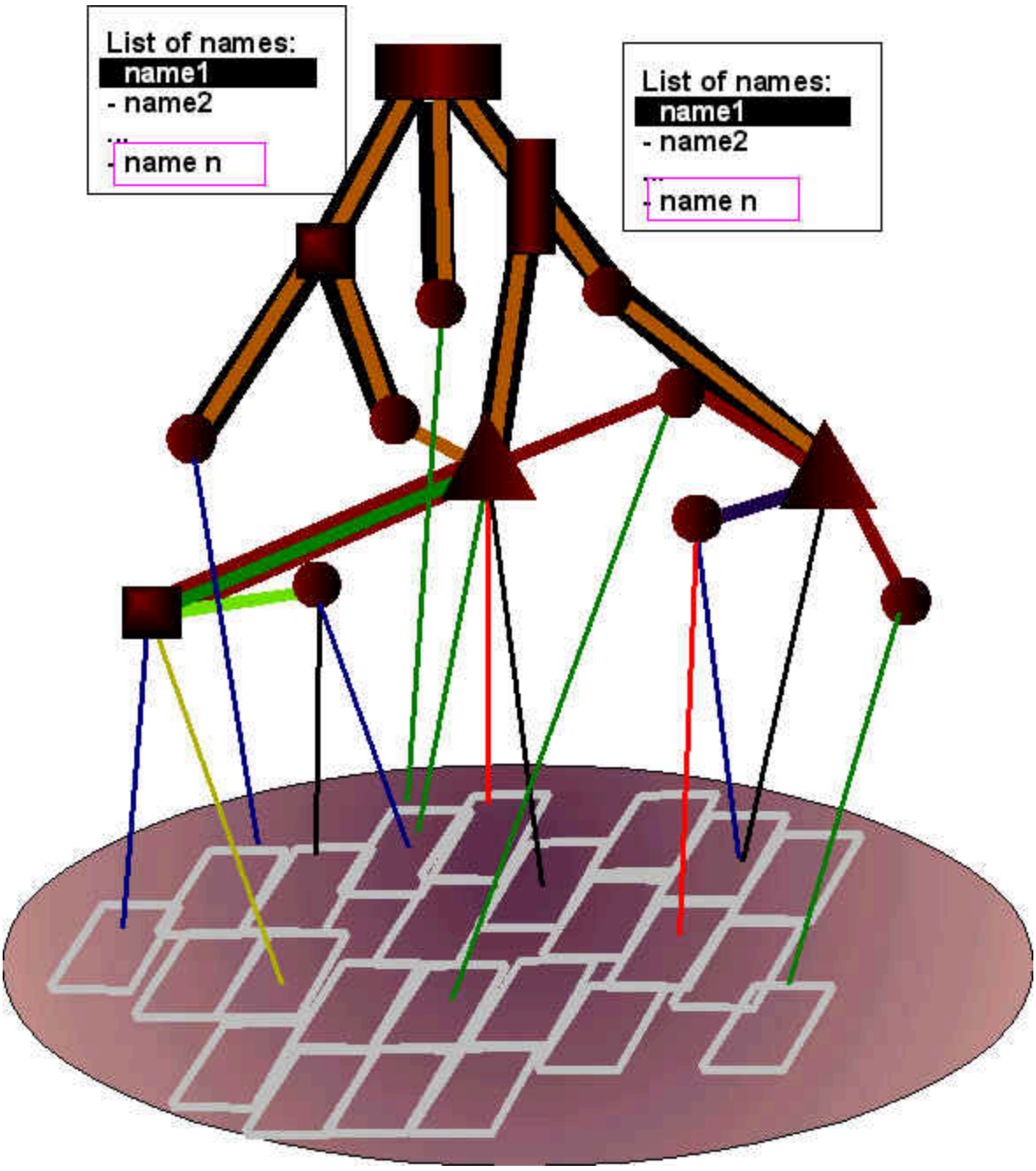
In other cases, such as encyclopaedias, selection criteria used for filtering can be chosen by the users who would have the possibility to check the list of criteria. they want to make visible, or to hide.

Topic map-based applications offer a variety of features to help topic map designers to decide where to apply filters, to provide them with ways to produce partial views for delivery, and to offer to users the possibility to tune the visibility they want, in case multiple levels exist.

The examples that have been given in this chapter are only examples of what can be done using filters. The nature of filters is left open to the topic map designer.

Scopes

As topic maps serve to describe knowledge statements, they are usually absolute, especially because they are intended to be used by computers. Sometimes, it may be useful to say that a topic is called with a given name within a certain context only, or that a given relationship applies only if one condition is satisfied. The Topic Navigation Map Standard has a way to nuance these affirmation, by using the feature called *scope* . Therefore a scope is used to limit the validity of an assignment.



Scopes applies at three different levels in the Topic Map standard:

- on name assignments
- on occurrence roles for topics
- on associations between topics

Scopes are described with one word, or several words, and the Topic Map Designer may choose to document what a scope means. In other words, a scope is itself a topic. The specific topics that are used to declare and optionally define scopes are called *scoping topics* .

Any assignment can be made within several scopes and not only within only one. The more scopes apply on an assignment, the more it is precise.

Implicit / Explicit Scopes

In some cases, scopes exist even when they have not been explicitely declared.

- Each given topic map has an implicit scope, which corresponds to the name of the given topic map. This scope, called the **unconstrained scope** , is the one that confers to all topic map elements their namespace. This feature is especially useful when combining elements originating from various topic maps, since it enables the possibility to keep track of the origin of each element. This unconstrained scope is the equivalent of the concept of *namespace namespace* used in XML. XML is the eXtensible Markup Language, a subset of SGML. Version 1.0 has been released in March 1998.
- The name of an occurrence role for a topic link is an implicit scope. This means that use of this facility triggers the creation of a scoping topic, whose full name has the value as declared as the value of the attribute occrole. This scoping topic should be understood as a placeholder, or a template, provided by the application to the user who is then free to decide to fill the space with a description of the scope.

Blanket added scope

Within a topic map, we are confronted to the "blind point", i.e. the fact that we can't see the position where we stand. We think our point of view is absolute, while somebody from outside will immediately recognize that it is only meaningful within a given context, which happens to be ours at the moment. In other words, the absence of scope for someone is his own scope, viewed by somebody else.

What precedes applies to topic maps. This issue will become visible only when mixing together topic maps from different origins. It will then be necessary to identify from which source each of them comes from.

This problem is similar to the namespace problem in the XML world. It has been solved by imposing to any information owner to declare an absolute name space, which will then be used as a prefix before the tag names. The problem with the XML namespace policy is that there are no ways to insure that two information owners will not use the same prefix for their information. Therefore, there is no way to guarantee with an absolute certainty what is the origin of a given information, or rather of a given tag set.

While using topic maps from different origins, the problem is the same. How can we ensure that a given topic originates from a well identified topic map source. If we were using a mechanism similar to the namespace in XML, the problem will be identical.

In TopicMap Land, there are two ways to state what is the origin of an information:

- by putting a name on every element which participates in the topic map document.
- by putting an attribute on a topic map document element, saying that all elements contained in this document is qualified by a given origin.

As the qualifier for the origin of a topic map provides a context, it is called a scope. And it is defined like other scopes within a topic map, i.e. by means of a topic element. This scope is special, and therefore has a special name. It is called the "unconstrained scope" or "unconstrained name space" or "blanket added scope".

Blanket added scope attribute

Scoping topics that were added to the scope of every topic characteristics in the whole document, using the blnkscop attribute of the document element, are implicit scoping topics.

Blanket added scope data attribute

Scoping topics that were added to every topic characteristis in the whole document, using the blnkscop data attribute of the document's entity declaration in the hub or subhub document entity to be a member of the BOS, are implicit scoping topics.

Defaulting Mechanisms

Defaulting mechanisms apply to the following topic map constructs:

- Topic Type (`type`)
- Occurrence role name (`occrole`)
- Occurrence scope (`scope`)
- Association type (`type`)
- Association role name (`anchrole`)
- Association role type (`type`)
- Facet Type (`type`)
- Facet value (`fvalue`)
- Facet value Type (`type`)

Public Topics

Topics are the building blocks used in topic maps. Topic maps can therefore be seen as sets of topics and associations. Since topic maps are usable for a community of users, it is therefore useful to be able to register certain topics, that can be used by reference. Declaring a public topic can be done by a registration authority, but also by any user, who decides that a particular topic or set of topics deserves to be considered as public, for future reference. Reference to a public topic is similar to the reference to a published item.

A public topic can be "organized", as any other topics, as a set of names followed by a set of occurrences. There can be several names for each public topic. And there can be as well several occurrences, including for example, description, definition, photograph, etc. The names of the occurrence roles are not enforced by the fact that it applies a public topic. However, once a public topic is referenced, all occurrences are part of the reference.

Public topics can be used for:

- registered trademarks,
- Internet domain names and name spaces
- product names
- countries, states, cities, etc.
- bibliographic references (books, URLs, etc.)
- and more.

Lists of topics are created and maintained by authorities, that operate as a dictionary of terms valid industry-wide, nation-wide, or on any scope that can be thought of. Two steps are involved:

- Registering public topics
- Referencing public topics.

Registering public topics

By refering to a given topic defined by an authority, it is possible to remove any ambiguity in the use of a given term. Therefore, it should be possible in a given topic map, to point to existing topics rather than having to create them from scratch. The topics that are available for re-use are called "public topics".

A public topic is defined by means of a public identifier, with the following format:

```
registration indicator, "//", topic authority, "//", public text class,"TOPIC", namespace of topic names, ":", topic name
```

The registration indicator, public text class and language fields are as specified for public identifiers in SGML (ISO 8879:1986).

The topic authority is the owner of the information resource that defines the topic.

The "TOPIC" keyword indicates that the formal public identifier specifies a public topic.

The "namespace of topic names" identifies the resource in which the "topic name" is the unique name given to the concept.

When a value is specified for the "public" attribute of a topic element, it is the name of a public topic entity.

For example, a topic which is a product of an unregistered company might be referenced as follows:

```
<!ENTITY gadget1 PUBLIC "-//XYZ Corporation//NONSGMLTOPIC
1998 Product Catalog Number: 43X2432//EN">
...
<topic id=id1 public=gadget1>
```

Here, the registration indicator is "-". It means that this authority is unregistered. It would be + if it would have been registered.

The topic authority is "XYZ Corporation".

The public text class is "NONSGML TOPIC".

The namespace of topic names is "1998 Product Catalog Number".

The topic name is "43X432".

The language is "english", abbreviated in "EN".

Beyond the Link Model

The Topic Map Model has been first designed and created as an application for the elaborated link structure created in HyTime, and enhancing features of SGML.

The implementations and topic map realizations have shown that the link structure was operational, but the work of designing the topic map architecture has also highlighted the limits of the link model.

The link model actually comprises two distinct representations:

- The connection between addresses. In the most usual case, one object is the source of the link and contains the address of another object. By triggering this link, the software application displays the target object. This is similar to cross-references, where the indication of page number is an invitation for the reader to turn the pages until finding the target page. The case of bibliographic references is very similar, except that it is necessary first to find another document before opening it at the right page. The source document need not be represented by an address, because the object itself serves as a source of the link. By enhancing it to an address, it makes the link more consistent, and reversible, if needed.
- The representation of a semantically-loaded relationship between a number of objects. A link describes a relationship, which has several aspects. Each aspect is called the role played in the relationship. Each role defines a group of anchors, i.e. the objects that are participating to this given relationship with the same semantics. The way they participate to this relationship is

marked by the fact that their address is reported as a member of the group for each anchor role. This model clearly distinguishes linking semantics from addressing.

The HyTime model imposes that the semantics of the link be well defined, i.e. that the nature of the aspects, or anchor roles, of each group of anchors be pre-defined, in an explicitely declared structural definition, which obeys to the syntax of document type definitions defined in SGML. This feature imposed fixed anchor roles.

This model fits very well to the application where there is centralized control over the nature of the links. But it does not fit the case where links can be defined on the fly, by adding new link types that have not been pre-declared by the topic map designer.

In its 2 nd edition, published in 1997, HyTime contains a new architectural form for links, called *variable links* (`varlink` in the HyTime jargon), which precisely allows to leave the role played by anchors free to be defined by a topic map user at any time, without breaking the definition for links formally defined in a DTD.

However, this model has a drawback, i.e. it does not allow to pre-define a number of possible anchor roles to which others can be freely added. More precisely, it doesn't make possible to define anchor roles that depend on a given link type and to which others can be freely added.

Another problem that has been revealed during the design of the Topic Navigation Map standard is the fact that the semantics of the link is entirely expressed in the value of the anchrole attribute. This is not sufficient to fully describe the semantics of what this anchor role means. Therefore, it would be more useful instead to provide an indirection to a topic, that can fully describe what this anchrole property means for that particular link.

By doing so, Topic Maps become a general multi-purpose documenting mechanism allowing meaningful annotations to be created on practically everything. In that perspective, a topic can be seen as a giant bookmark, and the topic map becomes then the collection of all these bookmarks.

Two points of view

A link centrally defines a complex list of objects, represented by their addresses, which participate in the relationship. Each of the object linked through such a link does not need to be identified as an anchor. It is only after having been processed by a link engine that the anchors are recognized.

Rules apply to define what is the application scope of a link. For example, the fact that if a link is made to another link, it also applies to all its anchors.

An equivalent, decentralized point of view consists in marking all anchors one by one, as anchors for a particular link. In this perspective, the link emerges as the set formed by anchors relative to the same semantics.

Rules apply as to how the grouping is made.

The two points of view are equivalent if and only if the declaration AND the rules to apply them match with each other. The rules applying to each of these representations can be called "link operators" and "link constructors". Link operators smash properties defined in one place in the set of locations declared. Link constructors collect all the information interspersed in a given information object set (defined in a BOS) and make links and other common properties emerge after processing.

This discussion shows that for a link model to be complete, it is not only necessary to have a description of where and what is connected together through a link, as it's the case in the traditional (even revised) HyTime/XML representation, but it is also necessary to express in a clear and unambiguous way the rules applied to assign properties through links. Actually, the problem comes from the fact that when a link is applied to an object, nothing in the HyTime standard says explicitely what rules should be followed when applying this link, regarding for example subelements of the objects. In one application, it will be important to have a

link apply on all subelements of a given object as well, while in other applications, the "anchor property", i.e. the fact that an element is an anchor, will not be inheritable.

How to represent a topic

Topic with a unique name

Here is an example that represents a topic: the city called "New York City".

First step : declare "topictype" as the name used for the public topic "TOPICTYPE".

```
<!ENTITY topictype    PUBLIC "ISO/IEC 13250:1999//NONSGML TOPIC Clause 6.1.2.3: TOPICTYPE//EN"
>
```

Second step : declare "city" as a topic type. "City" is a the name of a topic which is declared as conforming to the public topic "topic type", and therefore becomes a topic type.

```
<topic id="pubcity" public="topictype">    <name><fullname>city</fullname></name>
</topic>
```

Third step : Declaring New York City as the name of a topic belonging to the type "city".

```
<topic id="nyc" type="pubcity">     <name><fullname>New York City</fullname></name>
</topic>
```

Summary:

```
<!ENTITY **topictype**    PUBLIC "ISO/IEC 13250:1999//NONSGML TOPIC Clause 6.1.2.3: TOPICTYPE//EN"
>
<topic id="**pubcity**" public="**topictype**">
    <name><fullname>city</fullname></name>
</topic>
<topic id="nyc" type="**pubcity**">
     <name><fullname>New York City</fullname></name>
</topic>
```

It is possible to use the name of the topic type as the generic identifier of the topic

```
<topic>
<city id="nyc">
<name><fullname>New York City</fullname></name>
</topic>
```

This mechanism will find the topic "city", because the generic identifier is the same of one of the string of characters used as one among several possible full names used for a topic.

```
<!ENTITY **topictype**    PUBLIC "ISO/IEC 13250:1999//NONSGML TOPIC Clause 6.1.2.3: TOPICTYPE//EN"
>
<topic id="pubcity" public="**topictype**">
    <name><fullname>**city**</fullname></name>
</topic>
<city public="**city**" id="nyc">
     <name><fullname>New York City</fullname></name>
</city>
```

The tag "name" can be omitted. This example becomes:

```
<!ENTITY topictype    PUBLIC "ISO/IEC 13250:1999//NONSGML TOPIC Clause 6.1.2.3: TOPICTYPE//EN"
>
<topic id="pubcity" public="topictype">
    <name><fullname>city</fullname></name>
</topic>
<city id="nyc">
     <fullname>New York City</fullname>
</city>
```

Topic with two names

Here is the same topic with two names

```
<city id="nyc">  <name>
     <fullname>New York City</fullname>
     <fullname>New York</fullname>
     <fullname>Big Apple</fullname>
  </name>
</city>
```

In order to distinguish in which case the various names should be used, the scope attribute can be used on names. Here we are going to use three scopes:

1. Formal
2. Usual
3. Colloquial

These three scopes should be created as topics.

```
<topic id="sFormal">   <name><fullname>Formal</fullname></name>
</topic>
<topic id="sUsual">
   <name><fullname>Usual</fullname></name>
</topic>
<topic id="sColloquial">
   <name><fullname>Colloquial</fullname></name>
</topic>
```

Then, the names can be declared to be applicable within a given scope:

```
<city id="nyc">  <name>
     <fullname scope="sFormal">New York City</fullname>
     <fullname scope="sUsual">New York</fullname>
     <fullname scope="sColloquial">Big Apple</fullname>
  </name>
</city>
```

Foundations

The Topic Map Architecture is based on an underlying model, which is on the cutting edge of information technology modeling, and which inherits of the conceptual breakthroughs brought by the Standard Generalized Markup Language, and its hypermedia extension, HyTime.

Architectural forms

- Architectural forms are templates for DTDs
- Most of them mainly contain attribute lists that can be applied on any element, or elements of a specified type.

Architectural forms are a flexible, and powerful means of interchanging non-constrained structures, which add to the descriptive power of SGML and derived standards.

SGML has been designed as a metalanguage for building structured documents. An SGML document is made of three parts: an SGML declaration, which contains information to configure the system, an SGML document type definition, which contains the declaration of the markup being used, the nesting of elements and subelements, as well as qualifiers for elements, called attributes.

The more SGML applications develop, the more it becomes necessary to map between various structures or substructures. An SGML DTD is not sufficient because it does not provide the flexibility necessary to assign common features to document elements described with different DTDs. SGML architectures provide a mechanism to map various structures. They can be viewed as templates for document type definitions.

An SGML architecture is made of a meta-dtd. A meta-dtd is a set of architectural forms. An architectural form is a template that indicates how to instantiate derived element types or attribute lists in DTDs. There are three types of architectural forms: entity [?], attribute list, and element type.

Architectural forms can serve as a set of properties grouped together. They are most often expressed as a list of attributes. When an element declaration is instantiated within a DTD, it "inherits" the properties defined in the architectural form from which it is derived.

Topic map constructs are made of architectural forms, which in turn inherit from HyTime architectural forms for links.

Architectures can be divided into two categories: base architectures and derived architectures. This approach enables designers to modularize the description of structuring markup. The Topic Map architecture is derived from the HyTime architecture.

Features of architectures

Architectural Form Name

This is the name used throughout the instances of documents which use this architecture. This name is not used in the architecture itself. Instead, it is declared once for all in the *Architectural support declaration* .

Renamer

The elements that are defined in an architecture do not have to be used as such in the client documents. It might be convenient to use other names than the one declared in the meta-DTD, for example if one has to do with legacy documents, containing already element names. This feature allows users to remap names used locally to names used in the architecture. Thanks to this facility, various documents which use different names for elements can all be mapped to the same architecture.

suppressor

When architectural processing is performed on a set of information objects, there might be conflicts between the way the architecture is expressed and the structure expressed by the architecture. Therefore it might be convenient to suppress architectural processing for some of the elements.

Ignoring Architectural Processing

Document Element Form Name

Meta-DTD

Quantity

Data Form Name

Bridge

The bridge forms are used to do something with the elements which have no architectural counterpart. Doing something means for example applying to them architectural attributes that can be applied everywhere, on elements that are part of the architecture as well as on elements which are not part of the architecture.

Automatic Form Mapping

Options support

Document Management Features

The notion of Bounded Object Set

Groves and Property Sets

A grove describes the representation of a structured information object set after it had been parsed. It's not only a tree, but it's a tree of trees.

Topic Maps for the Web

Introduction

Topic Maps are well adapted to the Web. Although not limited to the Web applications, they can be used profitably to help Web users design pages which are easier to retrieve.

RDF

There is some common points between the Resource Data Framework (RDF) Model and Topic Maps that need to be further studied in more detail.

W3C efforts

Topic Maps can be used to help solving the namespace problem.

Xlink

Topic Maps are the first application of XML Extended Links. The ISO/IEC 13250 standard uses the HyTime varlink syntax, which is very close to the XML Extended Link syntax.

Topic Map Software

Overview. Topic Map Editors, Topic Map Engines, Topic Map

Navigators

- They are hybrid between document-based and database-based software applications.
- Search/Indexing engines can be used as front end to automatically create topics and relationships on top of unstructured information.
- Topic map editors allow their users to customize such topic maps, or create them from scratch.
- A Topic Map engine understands the standard notation of ISO 13250 and is able to import and export this information to a system able to process topic maps.
- Topic Map Navigators are designed to reflect the structure of the topic map: topic types and instances, topic relationships types and instances, occurrence roles, filters. User interface may vary from one product to another one. Formats used for navigation may vary as well.

Topic Map Navigation

- Displaying the structure of the topic map for non-Topic Map aware users.
- Example: Web interface, with "topic map screens" automatically generated.

- Example of use of filters: Customizing information types with Microsoft HTML Help browser.

Extracting Semantics from structured documents

- A given element instance can be considered a topic with a given type being its element type.
- Example: `Paris` , *city* becomes *topic type* while Paris becomes topic instance.
- Possible extraction in-context. Use of SGML/XML based software to extract relevant topics and topic types.
- Topic relationships can be created by scanning the document instance, i.e. two consecutive elements can be considered related together.
- Same mechanism applicable to database fields in given records.

Extracting Semantics from Unstructured Documents

- Use of sophisticated research engines can help creating topics and topic relationships (types and instances).
- Automatic indexing based on predefined lists of strings to be considered as topics.
- Any other specific processing for customizing an application (same as conversion algorithms).

Text Mining, Database Queries and Topic Maps

There are basically two ways to search through information: within information which is already structured, stored in databases or in structured documents, and within information which is not structured.

Full text retrieval

Search engines are designed to search and retrieve information within unstructured sources. They are based on various implementation choices that are efficient in given domains, or have general scopes. Products differ by their speed and efficiency. Some of them have built-in thesauri in one or several languages. Search engines belong to the domain of "text mining", which, associated with "data mining" represents an important trend in the field known as knowledge base engineering.

On the opposite sides of the scale of mining tools, full text indexing is an extensively used feature. Full text indexing is a technology which builds indexes will all words found in the documents being indexed, sometimes excluding words that have no semantic value, such as articles, prepositions, and the like. They can be used before browsing, in order to enhance speed, or they can be run on the fly, and return the words as currently present, at every moment. Full text indexing is a great technology to search for words that are known to be unique, such as technical or scientific terms, or product code numbers. But it often returns too many hits, and the result is difficult to exploit.

Other search engines are based on a built-in thesauri. The engine is able to parse every sentence and recognize the grammatical structure of it: verbs, adjectives, nouns, etc. Some of them include one, or several, built-in thesauri which contribute in building a semantic networks between terms used. They are able to recognize synonyms, and create a navigation which is more elaborate than with full text indexing. Every product has its own way to add features to these navigation devices, and users have to follow the rules that are specific to each of them.

It is not always possible for users of search engines to customize the hits returned by a search process. This means that it won't be possible to eliminate the noise generated by an exaggerated number of matches. Each time somebody will look for a given key, he or she will have the same problem to solve.

Database queries

On the other side of the spectrum, structured information bases can be navigated using the structure they contain. A database can be searched using queries based on combinations of values for various fields. Boolean operators are usually available, and enable users to search exactly for what they want. SGML/XML document management systems have similar features. They are able to retrieve elements depending on what value they exhibit. These two cases are very similar. They enable retrieving content from the structure. The tools that are built on this principle are not as easy to use than the ones based on full text search. The user needs to learn how to use the interface for the query language that is provided with each product. But more important is the fact that any user has to know the structure before he can issue a query. Because databases and SGML/XML structured applications do not require the structure to be fixed -- fortunately -- each application is therefore different. A librarian, for example, will have to learn how to query catalogs that use a MARC format, while a user of a customer database will have to learn how to query in this specific database.

On the one hand, users depend on the features provided by the software they have chosen. They are not able to decide by themselves what principle they are going to use in order to retrieve a particular set of information objects. On the other hand, users are obliged to follow the existing structure, even if it doesn't correspond exactly to the requirements they have for navigation. And they will have to adjust each time to the specific structure they are using.

Metadata

The most frequent case is when the two modes are mixed. The structured portion of unstructured information usually called "metadata", is common to all kinds of information sources, whether structured or unstructured, and can be used as handles to the information sources. Metadata are a very old and very classical way to minimally structure information.

Well known metadata are the title pages of books, which are used as foundations for building library catalogs. The metadata are structured in a way that is supposedly unambiguous, and each of a particular set of metadata serves as an entry for a field in a database. For example, library catalogs all have a field for "author", a field for "title", a field for "publisher", etc.

Metadata are also the bridge that relates document-centered approaches with database-center approaches. They are a minimal part of structured information added on unstructured information that serves to build a database of documents. Each book becomes a record in a database, and the metadata describing the books are the structured building blocks for the database of books.

The metadata approach has proved to be very efficient, and is now being extended to the information stored on the Web. The RDF - Resource Data Framework - specification under definition is an example of such a global approach. The Dublin Core comprises a minimal set of metadata that are useful to identify online information. Librarians think that these approaches are generic enough to represent a common core for the future standards to represent catalogs.

However, the drawback of the metadata approach is that it is necessary to agree on minimal cores of common fields that are going to be used universally. This is certainly the simplest and most natural way to envision it, but previous experience has proven that this works only as long as the information to be described is general and no specific descriptors are required. Once it is needed to be more specific in a given environment, people tend to disagree, and there is a limit to the number of described that can be added to a common universal core. This process has already been at work at the time of the GenCode project in the late 1970s and early 1980s, where publishers wanted to design a universal catalog of tags that could be used by everyone on the planet. The second famous try for this approach is the Hypertext Markup Language on the Web (HTML), which is good for minimal coding, but insufficient if more sophistication is needed. The discussion on metadata is of the same kind. It's not difficult to imagine that the same problems will arise in this context.

Topic Maps and search engines

Topic Maps can help improve use of search engines, database/SGML-XML queries and metadata.

Search engines of a new generation can be used to produce topic maps. Such topic maps, once built, will be under the control of the user, who will then be able to edit and customize the result of a search. Search engines can therefore be seen as front-end

applications for building editable topic maps, and are therefore some kind of computer-aided technology used to provide creation of topic maps from scratch, when no handle for topics and topic relationships pre-exist.

The same is true for mechanisms based on issuing queries on structured information bases. The act of writing a query can be something that is prepared by the author of a topic map only, leaving the resolved result for a topic map browser to display. (See Traditional navigational devices as resolved queries) For example, an index, a glossary, or a thesaurus are special queries that are issued on an information source set containing a link database. Instead of requiring end users to learn the mechanism necessary to create for example an index, the index query would be pre-defined, and the ease of use of the navigation would then be much enhanced. The query language could be made accessible only for advanced users, who do not find in the resolved queries the ones that he or she needs.

Structured repositories can work together with unstructured repositories using topic maps. Because a difference is introduced between the information itself which is needed for the topic map to operate and the process that has been necessary in order to produce the topic map at first place, the resolved topic maps are independent of the processes used to produce them, and therefore it doesn't matter for a topic map navigator what it was to begin with. Therefore, it is perfectly possible to mix in a topic map documents coming from unstructured repositories with documents originating from structured repositories.

This advantage is even more visible with metadata. Metadata can be simply extracted and serve as the foundation for topic maps. If various metadata schemes are used simultaneously, they can be reconciled at the topic map level. If the information repository contains information which possesses metadata as well as information without any metadata, then it is still possible to merge those together, by adding the information missing in the topic map because it could not be extracted from the metadata with the information coming directly from the metadata.

Traditional navigational devices as database queries

- Database queries are the traditional way to navigate information.
- SGML DTD-based queries are included in this category.
- This type of navigation requires knowing the structure.
- The query interface can be fairly complex.
- Navigation depends on the pre-existing structure: database schema, or SGML/XML DTD.

A major breakthrough of the conceptual work that has led to the Topic Map standard architecture is the fact that traditional devices, such as indexes, glossaries, thesauri, tables of contents and even cross-references can be seen as specific, resolved queries in a link database.

This perspective obliges to change the traditional vision of seeing these navigational devices. Especially the publishing way of seeing things has to be dramatically altered. It is one of the reasons why it took so long to arrive to a model that was considered satisfying.

Checking Topic Map Input

Open versus Closed environments

On the one hand, when topic maps are organized on a wide scale, it may be necessary to control the topic types and topic instances that can be created, or anchored, as well as the association and facet types and instances.

The topic map syntax can be used profitably to constraint the types that can be used within a topic map and regular SGML parsers can be used to check if the resulting topic map is correct.

On the other hand, an individual user, or a small group may want to create and populate topic maps in a completely open environment.

Fixed versus unfixed anchor roles

In HyTime, the architectural forms hylink and ilink have fixed anchor roles. In other words, it means that each combination of anchor roles gives rise to a different element type, and must be declared as such in a document type definition.

By contrast, the varlink architectural form does not constrain the anchor roles, which can be freely be declared by the user in any instance of the link element, and offer therefore more flexibility, while allowing less control. However, a varlink can be

Use of SGML editors versus HyTime engines. Varlink can be fixed in the DTD.

Topic Map Applications

User requirements

The motivation for designing the topic map standard comes from the need to make sense of the ocean of online information in which we are immerged. This requirement is particularly critical for environments where it is important to access rapidly, for many different reasons, the information which is asked for, and only this one. This chapter lists a number of situations where use of topic maps provides a significant advantage.

Electronic Library

Description : Multiple, heterogeneous documents stored and/or cataloged online.

With Topic Maps : Improve access to document titles, and inside documents. Provide powerful user interface to the catalog for non-computer expert users (not necessary to learn a query language). For librarians: enhance standard markup formats with customized fields. Apply several classification schemas.

Model : Topics include persons, departments, projects, concepts, products, articles, etc. Associations describe who is in charge of a project, what are the work positions of each person, who has written which project, etc. Facets can be used to use this information either as a management tool, or as a reference library, for example.

Software documentation

Description : Software documentation is frequently updated and some of it is re-used in new releases. Software documentation must be managed by synchronizing online contextual help, with global online documentation, and printed documentation.

With Topic Maps : Creation and maintenance of indexes, and glossaries applicable to diverse information sources, including software documentation from the manufacturer and books written by various authors. Reuse of existing documentation if pre-exist in different formats. Modularize documentation adapted to localization constraints (language) as well as to user profiles (experts, beginners, etc.)

Model : Topics include functions, variables, objects, methods, software component, etc. Each topic type is linked with its definitions, and other occurrences, that may be further specialized. Lists of topics can be provided to authors, even outside the software company, to write indexes and glossaries that are consistent with the ones coming from the manufacturer. Associations include some kind of relations between objects, or inclusion, or use. Facets help to differentiate between beginner and expert level. They can also be used to differentiate between languages in which documentation is produced.

Genealogy

Description : Creation and navigation of family trees. Takes into account various kinds of relationships between people, and must be updated regularly.

With Topic Maps : Complex relationships can be created. Information on the relationships between people coexist with records where the source information can be found. Several views can be derived from a Topic Map based genealogical information base.

Model : Each person is described as a topic, and is related to other persons through a variety of associations that describe the ways they are related. This exploded view of a genealogical tree enables the creation of multiple views (either describing the descendents of a given person), or looking for all relatives of a given person, etc.

Common thesauri for companies belonging to a consortium

Description : When many subsidiaries are part of a group, it's usually difficult to interchange information, because the way it is managed may differ from one subgroup to another. This situation becomes even more difficult when the subsidiary companies are spread out all over the world, because they can also use different languages.

With Topic Maps : Improve consistency between different companies by relating multiple existing thesauri, indexes, etc. This increases the overall value of the information owned by the consortium, which represents usually a critical asset for the company as a whole. Because Topic Maps users need to be explicit in what they are doing, it makes semantic interchange much easier (in addition to facilitating information between heterogeneous formats).

Model : The thesaurus is organized as a set of topics, related to each other by a number of associations that are defined to be consistent within the given information set. Depending on the environment and the actual requirements, this set of topics and associations can be either fixed or left open. Facets may help differentiating between knowledge domains, user profiles, access levels, languages and allow customization.

Unified views of heterogeneous documentations after companies

merge **Description** : Two companies are merging, and the resulting company has to deal with different formats, structures, and traditions for the documentation of the products. The next product line should take into account information that is derived from both sources, and retrofitting one into the other is not always possible, because there might be information that don't fit into the other model. Cost of retrofitting is huge and there is a need to reduce it.

With Topic Maps : Unify views of heterogeneous documentations after companies merge, by using topic map architectures able to point and retrieve to existing legacy documents. Designing appropriate topic maps should allow to maintain the existing document repositories in their current state, without requiring to retrofit one into another.

Model : Mapping between already existing structured models by relating existing constructs (for example, SGML elements) with corresponding ones at the topic map level. Existing element types can be considered as topics, and they can be related with associations. Facets may be used to refine the validity of the associations. This mapping, done at the model level, can be further extended to concurrently manage instances of documents.

Multi-lingual document management (e.g. European Community)

Description : Companies are selling products worldwide and need to deliver documentation in different languages. Or international institutions need to produce reference documents in several languages. Some information objects, such as graphics, tables, etc., could be re-used. There is a need for synchronizing the creation, and maintenance of these document repositories.

With Topic Maps : Links can help authors to manage workflow during the creation process. Topic Map facets are used to extract portions relevant to a given audience, e.g. by selecting documents in a given language.

Model : Each piece of information (e.g. a law article, a product, a component, etc.) becomes a topic. Consequently, its descriptions, mentions, etc., can be identified through occurrence roles, and a topic map can track the current state of creation of the current state of each of the relevant documents in different languages. During creation phase, separate topics can be instantiated for each language, and they can be related through associations that enable navigation from one language version to another. During delivery phase, all these topics can be grouped into unique ones, in multiple languages, language becoming a facet used for either filtering or differentiated between deliverable versions.

Legal information

Description : Legal information is heavily cross-referenced. At the same time, it needs constant update to reflect the creation of new laws, discarding of old one, taking track of the applicability issues, and connecting cases, and jurisprudences to reference texts.

With Topic Maps : Topic Maps can help navigate using meaningful topic structures, based on either identified application domains, or laws. Various associations are used to associate laws with their predecessors and/or successors. Facets are used to differentiate between audiences, and to divide the global information repository into categories such as Civil law, fiscal law, international law, commerce, etc.

Model : Laws, cases, jurisprudences, courts, concepts, countries, can be all considered topics. There are many associations that can be created between them, e.g. a law has been applied to solve a case, a jurisprudence is associated with a law, etc. Each law has an occurrence that refers to the text, and other occurrences that refers to places where it has been mentioned, or used. If necessary, these occurrences can be differentiated by their roles. Facets can be used to separate application domains, or law domains, or applicability areas.

Financial information

Description : Financial information, to be useful, must be presented in a way that contains all necessary connections to help take decisions quickly and reliably. Analyzing tables and comments can take a significant amount of time.

With Topic Maps : Accelerate access to critical information for executives who are sometimes -- or can be -- computer illiterate. Topic Maps can help authors prepare information in multiple ways that can help various targeted audience types to have the possibility of browsing at a glance only portion of information that is necessary for them.

Model : Companies, banks, investors, stock exchange, various markets indices, can all be used as topics. They can be associated by geographical proximity, activity sector, etc. Facets can be used to differentiate perspectives depending on whom the planned topic map is for.

Pharmaceutical industry

Description : Before a new medicine is authorized, it has to fulfill a number of conditions and have been extensively tested. This gives rise to the creation of huge documents, that are not always easy to go through, considering their size. On the other hand, having documents that are not extensive enough can be dangerous, because they lack the necessary material that must be taken into account.

With Topic Maps : These huge documents can be organized to give rise to several extractions adapted to different audiences, each of them having a role to play in the decision process.

Model: Topics such as products, experiments, parts of the human body, etc., can be ways to enter the documentation. Facets can be implemented for various targeted audiences.

Technical documentation

Description : Technical documentation can be fairly complex, such as documentation for airplanes. They are made of several parts, built by different manufacturers. Specifications for documentation delivery follow standards (such as the ATA DTD), but these specifications may vary over time, and the parts of the device are still useful after a specification has changed. Furthermore, once the product is assembled, multiple and complex links have to be created in order to reflect where each part of the documentation is applicable. Documentation is critical, as well for driving the machine, or for maintenance, and unclear documentation might lead to disasters. Therefore, it's critical to design and use powerful mechanisms to have all the documentation under strict control.

With Topic Maps : Topic Maps can help locating portions of the documentation that may not be 100% consistent, even if no other system has detected any other problem while checking portions of it. In other terms, the assemblage is different from the sum of all parts. Topic Maps are well suited to the description of the assemblage. By doing so, Topic Maps also encourage technical documentation users to modularize portions, in order to increase level of control. The level of control of Topic Maps is higher than with systems that have to be created from scratch for each new application. Topic Maps will consequently result in improving the quality of software that is used for dealing with such cases, by providing a more extended common ground for all applications, thus leading to the creation of commercial software that will be customizable.

Model : The creation of the model depends whether the topic map is aimed to be used as a user manual, a maintenance manual, a training material, etc., or all together. Devices, components, manufacturers, etc., can be modeled as topics and a number of relevant associations can be designed and created for each of them. Facets can help differentiate between various uses, and can also be used to produce customized versions for specific customers (user profiles), and multilingual versions of the documentation.

Training material

Description : Training material is created using various sources, including text, graphics, video, sound. They need to be integrated smoothly and designed in a way to facilitate access and browsing. This is sometimes contradictory with the necessity of keeping all modules separate enough for them to be reusable.

With Topic Maps : Flexibility of information use is described by using Topic Map facets. It's therefore possible to reserve use of certain portions of information for specific usages, or specific user profiles, such as beginners, intermediate, experts.

Model : Concepts to be taught are topics. These topics are related to others by a series of associations, whose semantics depend on the actual context. Facets can help differentiated between several user profiles, such as beginner, intermediate, experts.

Enriched navigation for value added web sites

Description : Some web sites are providing extensive resources providing access to a number of sources, commercial information, product catalogs, or numerous links to other sites. Web sites evolve frequently, and it is not always easy to manage their evolution, by keeping track of the links, sometimes within portions of documents, that can help accelerate research of given subsets of information.

With Topic Maps : It's possible to organize navigation to emulate indexes, glossaries, and automate cross-reference creation within an extensive web site. Maintenance of web sites are made easier.

Model : Building a Topic Map basically means that each topic can have its own web page, showing all links starting from it: either related topics or occurrences. Therefore, the design of a Topic Map for a web site will heavily depend on what are the topics appropriate for a particular environment. The same holds for associations, and facets. Use of facets requires in this context using a browser able to let users customize the output.

Encyclopaedia publishing

Description : Encyclopaedias are published after a long-term publishing. Quality and preciseness of information is considered important to distinguish between different encyclopedic products. Organizing navigation within the related terms and building and indexing terms is a costly task, that is difficult to coordinate.

With Topic Maps : By making indexing rules explicit, managing encyclopedic information with topic maps is made easier. The flexibility of the thesaurus-based knowledge base that can be created with topic maps greatly facilitates creation and update of encyclopaedias.

Model : Topic Maps already exist in encyclopaedia that are printed or published on Cd-Roms. Each term defined in the index is a topic. The related topics and thesaurus are described using associations. Facets can be used, for example, in the case where the data owned by an encyclopaedia publisher are aimed to be published in various partial encyclopaedia.

Conference proceedings, Scientific Journal Publishing

Description : Conference proceedings are articles written by various authors, that can be quite heterogeneous in format and presentation, unless strict rules are applied.

With Topic Maps : Topic Maps help relate various articles dealing with the same information content, i.e. keywords. A comprehensive index can be built very easily. Other types of navigation are possible, such as by laboratory or organization, or by geographical location of the places where authors are located.

Model : Topics may include, in supplement of technical terms, persons, companies, cities, countries. Associations may include information such as "employment", "location", etc. Therefore it is possible to browse a topic map not only following the usual index of terms, but also starting with a given country or city, or seeing the persons who are employed by a company or a laboratory.

Archives

Description : Research materials contain many useful informations that can be looked into to retrieve the origin of a scientific discovery, for example. However, organizing archive is a very costly task, and it's often discouraging to start. The situation is similar for state archive, although they are often organized in a more formal way. Archive may contain published material (e.g., articles) as well as unpublished material, notes, memos, etc.

With Topic Maps : Archive materials can be used as anchors for a topic map, therefore allowing quick access to any given list of predefined topics, whatever form they originally appeared. Constraint: they all must be accessible on line (even in the form of scanned material).

Model : Persons, terms, historical events, landmarks, places, etc., can all be used as topics, if relevant. A number of meaningful associations in a given context can be created (such as "This person has played a role in this event".). A number of facets can be created as well (e.g., access rights to a portion of information, that may be reserved to a limited category of people), etc.

Email

Description : Email archive, especially when technical discussions are involved, are a very valuable source of information. But the quantity of them, and the fact that they can not be easily searched except for full text, limits the interest of their use. It's often the case to remember that some subject has been discussed in a previous email, which may even be stored on the local disk, without being able to find the place where this discussion takes place.

With Topic Maps : In the topics maps model, topics can be seen as giant bookmark that help navigating to all of the occurrences of a given subject. Therefore, setting a bookmark on a given subject in a given email message makes it part of the topic about this subject, and greatly facilitates retrieval of information.

Model : Subjects of interest become topics, e.g. technical terms. Relevant associations depend on the domain. A topic map can be created by webmasters to give access to the archive, or it is also possible for each individual user to create his/her own topic map, reflecting a personal view on how this information may be used.

Product catalogs for electronic commerce

Description : Electronic commerce through the web requires presentation of catalogs of products, including descriptions, prices, availability, etc. Catalogs are constantly changing to reflect current state of the offer. Sophisticated catalogs on the Web are hard to maintain, especially when the pages are stuffed with links.

With Topic Maps : Topic Maps help organize and customize navigation, in order to provide to visitors the easiest possible access to the information made available. Easy maintenance is a key issue here.

Model : Products are obvious candidates for topics. They might be accompanied by related products, through an association.

Web searching

Description : The amount of information available on the Web makes it difficult to find relevant information on a given subject. Each web site has its own organization, and providers of Internet directories have their own techniques to search information. Therefore, search results depend on the technology used by search engines and the results might be quite different from one engine to the other.

With Topic Maps : Topic Maps help providers of Internet directories to maintain consistency on the search criteria therefore provide a more reliable environment for finding information. Results returned by indexing engines can be used as computer-aided input for building topic maps, that can be further customized to fill identified needs. Therefore, topic maps introduce a new competitive layer to differentiate between good and mediocre search engines, and web hubs. It's an issue that is wished by an immense number of Web users.

Model : Knowledge domains are good candidate for topics, as well of activities. Actually, so-called Topic Maps already exist and are created and are made available by Altavista. The Yahoo directory can also be used as a base for main topics. Combination of topic maps with search engine based on interactive queries should enhance results of web navigation.

Cross linking of library classification schemes and library

navigation systems **Description** : Several classification schemes exist for organizing library catalogs: among them, the Universal Decimal Classification, the Library of Congress Card Catalog, the Dewey Classification. These classifications have their own advantages and drawbacks, making sometimes difficult for a new library to choose which one to adopt. Sometimes none of them fits specific requirements for a given environment.

With Topic Maps : Topic Maps allow librarians to refer simultaneously to different classification schemas, and/or to build correspondence between the various existing schemas. Therefore, it enables interoperability between different classification schemas, even if they are based on very different premises. Eventually, if the various classification schemas are related at the highest level, it then becomes possible to create a specific classification scheme, which still relates to existing ones.

Model : Authority keywords in use by existing classification schemes are topics. Relations defined in thesauri used by the community of librarians form the basis of associations. Facets can be used to distinguish between classification schemas, and/or by language, and/or by knowledge domain. A classification system by facet already exists and is used by some libraries. More investigation should be made on this model to see if it fits with the topic map model.

Structuring of metadata schemas

Description : Metadata schemas are a set of fixed descriptors, generally inserted within the information sources themselves, that describe the basic bibliographic information about a given information object. The most current information available in metadata schemas include title, author, date, copyright owner, and optionally descriptive keywords. Namespaces are a solution currently being discussed to improve recognition of XML structural identifiers for navigating the net. They can be seen as the minimal basic unit of metadata schemas. Namespaces are created within one organization, but there is no provision available for documenting what the various names, or acronyms, will mean, and therefore no easy way to know what the information that is used within a

given context really means. Because of the limitations of namespaces, there is a risk that this approach will only work for a short time, until the growth of meaningful information interchange using XML will become in term difficult to manage.

With Topic Maps : Metadata schemas are used as references to automatically build topic maps. Topic Maps provide a way to organize namespaces within reference topic maps, that can point and document the various sources that are creating and maintaining namespaces. Each namespace can be documented. Equivalent alternative representations might be identified by associations, helping users to navigate between several namespaces.

Model : For example, RDF topics and subtopics become topics. Subtopics become full topics, related to higher level topic by a hierarchical relation. Dublin Core elements also become topics. Namespaces also become topics. All these can be related together, using facets to make only one representation appear, when useful. Documentation of namespaces can be easily added to a namespace treated as a topic. Navigation should be greatly enhanced by the fact that these all constructs can be built into a topic map (or, into as many topic maps as desired, if there is no agreement on what a particular topic means). Therefore, this should be a way to "unfix" some fixed schema, by adding another layer of description above it (without interfering with it).

Cross linking of structured metadata schemes

Description : Metadata are defined, and agreed upon, by a community of users. Various metadata schemas coexist, and there is no requirement that they provide consistency towards each other.

With Topic Maps : Because topic maps do not impose a given semantic to metadata schemas, it is therefore possible to use several concurrent metadata schemas simultaneously.

Model : See preceding model. Several different metadata schemes, even conflicting, can be made part of a unique topic map, that represents a customized view, able to reconciliate the various perspectives involved.

Topic type modelling

- First thing: differentiating topic types.
- Examples: person, company, product, city, artwork, function, etc.
- Topic typing can be also done by applying topic relationships on non-differentiated topic types (e.g. belongs to the type).
- Topic typing can also be done by filtering.
- Topic types can derive from pre-existing structures: SGML/XML elements/attributes or database fields.

Association Type Modelling

- Association can be different from topic types.
- i.e. Same relationships can be applied to relate topics belonging to different types, or
- Different relationships can be applied to relate topics belonging to the same type.

Facet Type Modelling

Multiple Simultaneous Topic Maps

- Several topic maps can be applied to the same information repository.
- For example, topic map for internal management versus topic map for external users.
- Topic maps are like views on a database.

- Topic maps can be related with other topic maps.

Distinguishing Occurrence Roles

- Occurrence role typing is done independently.
- Occurrence role typing can be used to further subdivize the semantics of the topic types.
- Examples of occurrence roles: definition, mention, photograph, graphic, biography, full description, main occurrence, etc.

Maintenance Issues

- Topic maps helps ensuring consistency within huge amounts of information.
- Maintenance of the Topic Maps is made independently of the evolution of the documents
- Hooks are used to point to information sources (application-dependent).
- Navigation reflects the changes in the Topic Map design or specifications.

Alternative Design Solutions

- Open model enables alternative design strategies.
- Decision depends on granularity level, semantic levels to be addressed, possible automatic processings versus manual coding
- Re-use of existing semantics (for example, SGML/XML DTDs) can greatly facilitate topic map design and management.

Merging Topic Maps

- Avoiding name clashes

Documenting a Topic Map with a Topic Map

TNM Syntax enables the creation of a document that documents the topic map, by listing all the constructs used in the current topic map. This document is not required for a topic map application to be processed. It is only useful in the following cases:

- when guidelines for the topic map design need to be interchanged, and therefore documented.
- when the topic map constructs have several names corresponding to different sets, and the user can choose which set to display. This case is particularly adapted in the case of multilingual topic map designs.
- when various topic maps are merged, and the maximum possible part of the merging process should be automated.

The topic map documentation is expressed in the form of a document described by a fixed DTD, which is a template to be filled with the specific constructs applying for a given topic map. Since there is a limited number of TNM constructs, a fixed DTD is appropriate to store the information documenting the Topic Map, and it will make the interchange of information on Topic Maps straightforward.

Note: This DTD can be used as a summary describing a TNM instance.

```
<!ELEMENT TopicMapDocumentation - -          (TopicType*, AssociationType*, FacetType*) >
<!ATTLIST TopicMapDocumentation
          TNM-namespace CDATA #REQUIRED
>
```

```
<!ELEMENT TopicType - -
          (Type, Scope+) >
<!ATTLIST TopicType
          id ID #IMPLIED
>
<!ELEMENT AssociationType - -
          (Type, Role+, Scope*)  -- Scope does not belong here. -- >
<!ATTLIST AssociationType
          id ID #IMPLIED
>
<!ELEMENT FacetType - -
          (Property, Value+) >
<!ATTLIST FacetType
          id ID #IMPLIED
>
<!ELEMENT Type - - (Name)+ >
<!ATTLIST Type
          id ID #IMPLIED
>
<!ELEMENT (Scope|Role) - - (Name)+ >
<!ATTLIST (Scope|Role)
          id ID #IMPLIED
>
<!ELEMENT Property - - (Name)+ >
<!ATTLIST Property
          id ID #IMPLIED
>
<!ELEMENT Value - - (Name)+ >
<!ATTLIST Value
          id ID #IMPLIED
>
<!ELEMENT Name - - (#PCDATA) >
<!ATTLIST Name
          id ID #IMPLIED
>
```

Topic Map Pointer Element

idloc, refloc, rflocsrc

Topic Map Documentation Element

A topic map documentation lists the topic map constructs that participate in the topic map. There can be zero or several topic types. The case of zero topic type corresponds to the situation where the topic map is used only for filtering, using only facets (and the facets are not documented as topics). The case of zero topic prohibits using any associations, since associations can only relate topics together, and therefore there must be at least one topic type in order to allow relations to be done.

There can be zero or more association types. Association types are only allowed to exist if at least one topic type exists.

There can be zero or several facets.

The Topic Map has a namespace attribute which distinguishes the names of the constructs in this Topic Map from the names of topic map constructs used in other Topic Maps. Even if the names of the topic map constructs are identical in two or more topic maps, they are considered to be specific to each topic map, thanks to the TNM-namespace attribute.

Topic Type Element

Each topic type has a type (and only one), represented as the content of a "type" element. More than one scope can be attached to a given topic type. Scopes are used to help differentiate occurrence roles for each topic.

AssociationType Element

An association type has a type (and only one), represented as the content a "type" element. A relation is made between sets of objects that play well-defined roles in the relation. Each role is declared in a "role" element. An association may contain only one role element, in that case the purpose of the association is to perform a grouping of objects sharing this role.

An association may be "scoped", i.e. it may be defined within certain limits that are made explicit. There can be zero or several scopes applies to each association.

FacetType Element

A facet type element contains a property and a set of values for that property.

Name Element

The element Type, Scope, Property, Value can have several names, corresponding to pre-defined sets, that can be selected for use in a given topic map.

Note: This facility has been designed to enable topic map constructs to be defined in multiple languages, and let applications display only the one corresponding to what the users want. The mechanism is more general and can be applied for any facet that is defined as part of the topic map documentation usage. See example below.

The name element contains characters.

Example

```
<TopicMapDoc><TopicMapDeclaration>

<TopicType id="topictype-painter">

<Type>
 <Name id="painter-painter">Painter</Name>
 <Name id="painter-peintre">Peintre</Name>
</Type>

<Scope>
 <Name id="painter-mention">Mention</Name>
</Scope>

<Scope>
 <Name id="painter-biography">Biography</Name>
 <Name id=painter-biographie">Biographie</Name>
</Scope>
</TopicType>

<TopicType>
<Type>
 <Name id="city-city">City</Name>
 <Name id="city-ville">Ville</Name>
</Type>

<Scope>
 <Name id="city-mention">Mention</Name>
</Scope>
```

```
<Scope>
 <Name id="city-streetmap">Street Map</Name>
 <Name id="city-plan">Plan</Name>
</Scope>
</TopicType>

<TopicType>

<Type>
 <Name id="construct-construct">Construct</Name>
 <Name id="construct-structure">Structure</Name>
</Type>

<Scope>
 <Name id="construct-mention">Mention</Name>
</Scope>

<Scope>
 <Name id="construct-definition">Definition</Name>
 <Name id="construct-deacute;finition">Deacute;finition</Name>
</Scope>

</TopicType>
<AssociationType>
<Type>
 <Name id="location-location">Location</Name>
 <Name id="location-localisation">Localisation</Name>
</Type>

<Role>
 <Name id="location-locatedin">Located In</Name>
 <Name id="location-situe">Situ&#233;</Name>
</Role>

<Role>
 <Name id="location-contains">Contains</Name>
 <Name id="location-contient">Contient</Name>
</Role>
</AssociationType>

<FacetType id="facettype-language">

<Property>
 <Name id="language-language">Language</Name>
 <Name id="language-langue">Langue</Name>
</Property>

<Value id=value-english>
 <Name id="language-english">English</Name>
 <Name id="language-anglais">Anglais</Name>
</Value>

<Value>
 <Name id="language-french">French</Name>
 <Name id="language-francais">Fran&#231;ais</Name>
</Value>

</FacetType>
</TopicMapDeclaration>

<!-- Topic Map created to document the Topic Map -->


<TopicMapDocumentation>
```

```
<Topic type=Construct>
 <Name>English</Name>
<Definition>English-def</Definition>
<Used-in>value-english</Used-in>
</Topic>
<p id=English-def>In this topic map, the word
English designates indifferently variants of the language used in the USA,
in Canada, the UK, New Zealand, Australia, South Africa, etc. When different
spellings for a word are possible, the American variant has been adopted.

<Facet type=language>
<English>painter-painter painter-mention painter-biography
city-city city-mention city-streetmap construct-construct construct-mention
location-location location-locatedin location-contains language-language
language-english language-french</English>
<French>painter-peintre painter-mention painter-biographie
city-ville city-mention city-plan construct-structure construct-mention
location-localisation location-situe location-contient language-langue language-anglais
language-francais</French>
</Facet>

</TopicMapDocumentation>
</TopicMapDoc>
```

Topic Map - A Historical Perspective

Historical Background: Davenport Group, CApH

- 1991-1992: Davenport Group. SOFABED: Unix vendors looking for interoperable indexes, glossaries and thesauri.
- 1993-1995: CApH (Conventions for the Application of HyTime): Looking how to apply powerful hyperlink concepts in HyTime to the navigation.
- 1996-1998: ISO 13250 (Topic Maps). International standard for interchange of Topic Map information
- The future: ISO 13250 compatible with XLL (Extensible Link Language), part of XML.

The Topic Map project, which will result in 1999 with the publication of an ISO standard, is the result of a collective work that has started in 1991.

Three years of intense brainstorming activity have been necessary until the model stabilized.

Two years have been devoted to early implementation, tests, consulting activities with customers in order to define real-world applications and prove the standard's concepts.

Three years have been necessary to improve the model while going through the ISO procedures, and transform the result of the previous work into an international standard.

Milestones

During the first period, the issue started with requirements from Unix systems vendors who wanted to improve the production and delivery process of their online documentation. Discussions were focused on a model that could answer the various requirements.

The second period started when it became clear that the hyperlink and location addressing features of the HyTime standard could be advantageously used. Internal workings for links have been studied at length.

During the third period, two major constructs have been isolated: Topics and topic relationships, both described as links. Topics can be used to describe indexes, glossaries and cross-references, while topic relationships can be used to describe thesauri, catalogs and tables of contents.

The fourth period was primarily marked by multiple faceted views of topic maps.

Starting point : The Davenport Group

The origin of Topic Maps can be traced back as early as 1991. Its history corresponds to at the beginning is the same as a group called the Davenport group.

The first meeting was held on May 28, 1991, and was a call for an Initiative for Online Documentation and Publication. It grouped participants mainly from the Unix vendor community, who were looking for ways to improve interoperability of the documentation for Unix systems. The issues discussed then were the necessity for interchange and interoperability. A solution based on SGML was advocated, and the example of the Document Type Definition (DTD) created by OSF (Open Software Foundation) was considered. The group is led by Dale Dougherty (O'Reilly && Associates).

On June 21, 1991, the second meeting of this group, hosted by Frame, decided for the name of "Davenport". The main issue raised at this meeting was the difference between the source format and the delivery format. SGML was elected as the best choice for interchanging source documents, but its use was deprecated as a delivery format.

During the next meeting, on July 16, 1991, the discussion continued on that issue. A comparison was made between SGML and Postscript, relatively to the source/delivery features.

In December 1991, HyTime is first considered by the Davenport Group, thanks to the presence of its co-editor, Dr. Steven R. Newcomb. A subcommittee was established to study HyTime, called DASH (Davenport Advisory Standard for Hypermedia). In January 1992, the discussions started on how to use architectural forms to describe navigational constraints in online documents.

In June 1992, a specification is created which is the direct ancestor of the Topic Map Standard. Its name is SOFABED, a specially coined acronym for Davenport, meaning "Standard Open Formal Architecture for Browsable Electronic Documents".

The structure of an index was discussed, as well as the difference between a master index for a group of documents and individual indexes for each documents. In parallel, another specification was created for studying versioning and access management: COUCH, meaning "Conventions for Online Use of Commercial Hypermedia".

In April 1993, members of the Davenport Group decided to focus exclusively on a document type definition for technical documentation, that O'Reilly started to develop together with HAL. This DTD is called DocBook and has since then given rise to various implementations in market-leading products. The members of the group who continue the navigational architecture created a new committee, CApH (Conventions for the Application of HyTime), under the auspices of the GCA-RI (Graphic Communications Association Research Institute). The group is chaired by Steven R. Newcomb.

Several meetings take place in the year 1994 and 1995 in places such as New York, Tallahassee, Montreux, Vancouver, Singapore, Washington, Rochester, Boston. They were usually coupled with the SGML conferences organized by the GCA.

In 1994 the structure of a topic as a link emerged, and represented the end of a brainstorming process seemingly endless, where new solutions, more appropriate, were found at each meeting, giving some of the members of the group the feeling that things were progressing in a circular way. The model was tested and implemented using a HyTime engine in a prototype developed at High Text and called HNS (Hyperdocument Navigation System), which was using TechnoTeacher's HyTime Engine, HyMinder.

Topic relationships arose as a supplement to topics that participate in the description of a Topic Map. The question where to include a fixed list of well-defined semantic relationships was discussed. Those which were considered include signifying (symbol-value), descriptive (item-definition), identity (item-identical item), equivalence (item-equivalent item), similarity (item-similar item),

opposition(item-opposite item), generic (item-related item), instance-class (instance-class), subset-superset (subset-superset), containment (part-whole), possessive (owner-possession), abbreviation (synopsis-expansion).

This list has been abandoned, because it was either too detailed, or not enough detailes. It is an illustion to believe that in such a standard we could express all possible relationship types that anybody would ever need. However, some of the relationships listed above can be used in a particular context.

The model of topic relationships has been extended, not only to any semantics a user wants to express, but also to any number of participating anchors. Topic relationships do not have to be limited to one to one relationships.

At the end of 1995 it appeared clearly that the potential of Topic Maps is too important to leave it as an unformal specification. The group agreed to propose to ISO the Topic Map Architecture as an international standard project and appointed Michel Biezunski for this purpose.

In Spring 1996, ISO WG8 adopts Topic Maps as a standard project. Michel Biezunski and Martin Bryan become co-editors.

Between 1996 and 1998, the CApH spec is progressively transformed into an ISO standard. A third dimension is added to the navigational features provided by topics and topic relationships: the possibility to define filters that are used to qualify information in order to include or exclude certain kinds of information, depending on the facet one wants to display.

Examples

A simple example

This example has been designed by Steve Newcomb.

In order to understand what's going on here, you have to understand the inheritance path. From most abstract to most concrete, here it is:

```
10744.mdt       (a miniature HyTime meta-DTD)
    |
13250.mdt       (a miniature TNM meta-DTD)
    |
mytopicmap.dtd  (a little TNM-conforming DTD)
    |
test.sgm        (a miniature TNM document)

The declarations work in the upward direction:

(1) test.sgm uses the markup declarations in mytopicmap.dtd as the
    bulk of the mytopicmap DOCTYPE declaration.

(2) The markup declarations in mytopicmap.dtd include declarations
    that cause the TNM meta-DTD in 13250.mdt to be regarded as a base
    architecture for the mytopicmap DTD.

(3) The markup declarations in 13250.mdt include declarations that
    cause the HyTime meta-DTD in 10744.mdt to be regarded as a base
    architecture for the TNM meta-DTD.

After you've edited the system addresses in the "catalog" file, you should
be able to run

(sgmlnorm|nsgmls) test.sgm
```

```
   This will output an ASCII representation of the grove resulting
   from parsing test.sgm against the mytopicmap doctype.

(sgmlnorm|nsgmls) -A topicmap test.sgm
   This will output an ASCII representation of the grove resulting
   from parsing test.sgm against the Topic Navigation Map meta-DTD.

(sgmlnorm|nsgmls) -A topicmap -A hytime test.sgm
   This will output an ASCII representation of the grove resulting
   from parsing test.sgm against the HyTime meta-DTD.  Note that you
   have to name the architectures on the command line in the same
   order that they occur on the inheritance tree, from the bottom up.
   (A little-known fact about SP.)

Note that <!AFDR only needs to appear in meta-DTDs that use the DTD
syntax extensions: #ALL (i.e., common attributes) and multiple
attlists for the same element type.  Because of this, in this example,
you must declare it in 10744.mdt, but it's not necessary in 13250.mdt
because I didn't use these extensions in that file.  I think we should
put it in the 13250 meta-DTD in any case, whether we use the
extensions or not.
```

The catalog file

```
PUBLIC "ISO/IEC 10744:1997//DTD AFDR Meta-DTD Hypermedia/Time-based Structuring Language (HyTime)//EN" "c:\\home\\srn\\TO
PUBLIC "ISO/IEC 13250:1999//DTD AFDR Meta-DTD Topic Maps//EN" "c:\\home\\srn\\TOPICMAP\\13250.mdt"
PUBLIC "-//Steve Newcomb//DTD A TNM DTD for demo only//EN" "c:\\home\\srn\\TOPICMAP\\mytopicmap.dtd"```
###
The ISO 10744 meta-dtd

<!AFDR "ISO/IEC 10744:1997">

<!--
ISO/IEC 10744:1997//DTD AFDR Meta-DTD
Hypermedia/Time-based Structuring Language
(HyTime)//EN

ABRIDGED!!!!!!!!!! for demonstration
of architectural inheritance by
Topic Navigation Map Meta-DTD
-->

<!--

Option Summary:

[General Architecture]

(none)

[Base Module]

bos       HyTime Bounded Object Set Specification Facility
valueref  Value Reference Facility (really not needed for demo)

[Location Address Module]

multloc   Multiple Location Address Facility (really not needed for demo)
nmsploc   Name-space Location Address Facility
refloc    Reference Location Address Facility (really not needed for demo)
reftype   Reference Typing Facility (really not needed for demo)

[Hyperlink Module]
```

```
varlink   Variable Hyperlink Facility

[Scheduling Module]

(none)

[Rendition Module]

(none)

-->

              <!-- Content model parameter entities -->

<!element
   HyDoc          -- HyTime document element --
                  -- Clause: 6.4 --
   - O
   (HyBrid)*
   +(varlink|nmsploc)
>


<!-- HyTime Architectural Bridging Element -->
<!element
   HyBrid         -- HyTime architectural bridging element --
                  -- Clause: 6.6 --

   - O
   (HyBrid)*

-- Attributes [base]: HyBrid --
-- CommonAttributes [base]: valueref --
-- CommonAttributes [locs]: refloc, reftype --
>

             <!-- Value Reference Attribute Namer -->
<!attlist
-- valueref --    -- Value reference attribute namer --
                  -- Clause: 6.7.1 --
   #ALL

   valueref       -- Value reference attribute namer --
                  -- Associates attributes (or element's content) with
                     referential attributes (or content) that can be
                     used to refer to the semantic value of the
                     attribute or content. --
      CDATA       -- Lextype: ((ATTORCON|"#ELEMENT"),ATTORCON)+ --
                  -- Note: The first ATTORCON is the attribute or
                     content for which the value is to be addressed,
                     the second ATTORCON is the attribute or content
                     by which the value reference is made. --
                  -- Constraint: second attribute in each pair must be
                     a referential attribute or content defined as
                     referential by the refloc facility. --
                  -- Constraint: "#ELEMENT" and "#CONTENT" may only
                     occur once as first keyword of pair.  "#CONTENT"
                     may only occur once as second keyword of pair. --
      #IMPLIED    -- Constant --
>


                <!-- Name-Space Location Address -->
<!element
   nmsploc        -- Name-space location address --
```

```
                  -- Clause: 7.9.3 --
                  -- Addresses objects by name in a name-space --
                  -- Constraint: location source must be an object that
                     exhibits a named node list property or whose
                     additional property source exhibits a named node
                     list property. --
   - O
   (#PCDATA)      -- Lextype: (word|literal)* --

-- Attributes [locs]: nmsploc, proplat, locsrc, impsrc --
-- OptionalAttributes [locs]: multloc, referatt --
-- CommonAttributes [base]: valueref --
-- CommonAttributes [locs]: refloc, reftype --
>
<!attlist
   nmsploc        -- Name-space location address --
                  -- Clause: 7.9.3 --

   namespc        -- Name-space --
                  -- Name-space property of location source from which
                     nodes are selected. --
      NAME
      #REQUIRED
>


                       <!-- Location Source -->
<!attlist
-- locsrc --      -- Location source attributes --
                  -- Clause: 7.2 --
   (nmsploc)

   locsrc         -- location source --
      CDATA       -- Reference --
                  -- Constraint: References to entities are references
                     to roots of primary groves constructed from the
                     entities. --
      #IMPLIED
>



               <!-- Multiple Location Attributes -->
<!attlist
-- multloc --     -- Multiple location attributes --
                  -- Clause: 7.4 --
   (nmsploc)

   ordering       -- Is ordering of locations significant? --
      (noorder|ordered)
      ordered

   set            -- Make multiple a set by ignoring duplicates --
      (notset|set)
      notset
>

                  <!-- Reference Element Type -->
<!attlist
-- reftype --     -- Reference type attributes --
                  -- Clause: 7.7.1 --
   #ALL

   reftype        -- Reference element type --
      CDATA       -- Lextype: (("#ALL",(GI|modelgroup|"#ANY"))?,
                              (ATTORCON,(GI|modelgroup|"#ANY"))*) --
```

```
                  -- Constraint: a given ATTNAME or #CONTENT can occur
                     only once; types apply to ultimate object of
                     address, not including any intermediate location
                     Address elements. --
                  -- Constraint: model groups limited to repeating
                     or non-repeating OR groups if refmodel option not
                     supported. Tokens in model groups must be GIs. --
      "#ALL #ANY" -- Constant --
>

            <!-- Variable Link -->
<!element
   varlink        -- Variable link --
                  -- Clause: 8.2.4 --
   - O
   (anchspec)+

-- Attributes [links]: ancspcat, varlink --
-- CommonAttributes [base]: valueref --
-- CommonAttributes [locs]: refloc, reftype --
>
<!element
   anchspec       -- Anchor specification --
                  -- Clause: 8.2.4 --
   - O
   (nmsploc)*     -- Reference --
                  -- Note: Use of refloc facility required --

-- Attributes [links]: anchspec, ancspcat --
-- CommonAttributes [base]: valueref --
-- CommonAttributes [locs]: refloc, reftype --
>
<!attlist
   anchspec       -- Anchor specification --
                  -- Clause: 8.2.4 --

   multmem        -- May anchor have multiple members? --
      (single|list|corlist)
      single

   emptyanc       -- Is empty anchor an error? --
      (error|noterror)
      error
>

<!attlist
-- ancspcat --    -- Anchor specification attributes --
                  -- Clause: 8.2.4 --
   (anchspec,varlink)

   anchrole       -- Anchor role --
      NAME
      #IMPLIED    -- Default: for anchspec, anchor role is GI of element --
                  -- Default: for varlink, varlink is not self anchor --
>
```

The ISO 13250 meta-dtd

```
<!AFDR "ISO/IEC 10744:1997">
```

```
<!--
ISO/IEC 13250:1999//DTD AFDR Meta-DTD
Topic Maps//EN

ABRIDGED!!!!!!!!!! for demonstration
of architectural inheritance by
"myptopicmap.dtd"
-->

<!-- This section declares that HyTime is a base architecture of this
     meta-DTD -->

<?IS10744 ArcBase HyTime>

<!NOTATION
   HyTime         -- HyTime Architecture --
                  -- A base architecture used in conformance with the
                     Architectural Form Definition Requirements of
                     International Standard ISO/IEC 10744. --

   PUBLIC "ISO/IEC 10744:1997//NOTATION AFDR ARCBASE
           Hypermedia/Time-based Structuring Language (HyTime)//EN"
>
<!ATTLIST #NOTATION HyTime
   ArcFormA       -- Architectural form attribute name --
      NAME        -- Constraint: must be unique among all attribute
                     names in DTD in which it is specified --
      HyTime      -- Default: ArcName --

   ArcNamrA       -- Architectural attribute renamer attribute name --
      NAME        -- Constraint: must be unique among all attribute
                     names in DTD in which it is specified --
      HyNames     -- Default: no renaming --

   ArcsuprA       -- Architecture suppressor attribute name --
      NAME        -- Constraint: must be unique among all attribute
                     names in DTD in which it is specified --
      sHyTime     -- Default: no suppression --

   ArcIgnDA       -- Architecture ignore data attribute name --
      NAME        -- Constraint: must be unique among all attribute
                     names in DTD in which it is specified --
      HyIgnD      -- Default: data is conditionally ignored --

   ArcDocF        -- Architecture document element form name --
      NAME        -- Constraint: name of element form in architecture
                     meta-DTD --
      HyDoc       -- Default: ArcName --

   ArcDTD         -- Architecture meta-DTD entity --
      CDATA       -- Lextype: (ENTITY|PENTITY) --
      "%HyTime"   -- Default: entity whose name matches the notation name
                     after SGML name and entity folding is applied. --
                  -- Constraint: a name must be specified or the meta-DTD
                     entity name after folding must match notation name
                     after folding. --

   ArcBridF       -- Architecture bridge form name --
                  -- For elements with an ID and no ArcForm attribute
                     whose form isn't defaulted --
      NAME        -- Constraint: name of element form in architecture
                     meta-DTD --
      HyBrid      -- Default: no defaulting --

   ArcAuto        -- Architecture automatic form mapping --
                  -- Automatic form mapping for identically named
```

```
                     element types and external data entity
                     notations --
      (ArcAuto|nArcAuto)
      ArcAuto

   ArcOptSA       -- Architecture options support attribute names --
      NAMES       -- Lextype: ATTNAME+ --
      "base locs links"

   base           -- Base module facilities --
      CDATA       -- Lextype: csname+ --
      "bos valueref"

   locs           -- Location address module facilities --
                  -- Clause: 6.3 --
      CDATA       -- Lextype: csname+ --
      "multloc nmsploc referatt refloc reftype"

   links          -- Hyperlinks module facilities --
                  -- Clause: 6.3 --
      CDATA       -- Lextype: csname+ --
      "hylink varlink"

>

<!ENTITY % HyTime PUBLIC "ISO/IEC 10744:1997//DTD AFDR Meta-DTD Hypermedia/Time-based Structuring Language (HyTime)//EN">


<!ELEMENT topicmap - - (topic)* >

<!ELEMENT topic - - ((name+, occur*) | occur+) >
<!ATTLIST topic
    HyTime   (varlink|HyBrid) varlink
    id       ID               #REQUIRED
    public   ENTITY           #IMPLIED
    type     CDATA            #IMPLIED
>
<!ELEMENT
  name            -- A name of a topic. --
  O O
  (fullname, dispname*, sortname* )
                  -- If dispname or sortname are not specified,
                     applications use fullname.--
>
<!ATTLIST
  name
   scope          -- List of references to scoping topics that
                     collectively define the limited contexts within
                     which the name is applicable to the topic. --
      CDATA       -- Reference --
                  -- Reftype: topic+ --
      #IMPLIED    -- Default: scope is unconstrained. --
>
<!ELEMENT
  (fullname | dispname | sortname)
   - O
  (#PCDATA)
>


<!ELEMENT
  occur           -- topic occurrence --
                  -- Clause: 5.2.3 --
  - O
  (nmsploc)+>
```

```
<!ATTLIST
  occur
   HyTime
      NAME
      #FIXED
      anchspec
   occrole        -- Occurrence role name. --
      NAME
      #IMPLIED
                  -- Default: occurrence role name is GI of
                     element. --
   scope          -- List of references to scoping topics that
                     collectively define the limited contexts within
                     which the occurrences are applicable to the
                     topic.  Each referenced topic may be regarded as
                     describing a class of occurrence.  --
      CDATA       -- Reference --
                  -- Reftype: topic+ --
      #IMPLIED    -- Default: The topic which has a name (within the
                     scope of the OCCTYPE public topic) which is the
                     same as the value of the occrole attribute, or,
                     if no value is specified for the occrole
                     attribute, the generic identifier.  Even if no
                     such topic exists in the topic map document(s),
                     it is considered to exist by virtue of this
                     defaulting mechanism. --
   linktrav
      (A|EI|ER|ED|EN|EP|ERD|I|ID|D|N|P|R|RD)
      A
   -- listtrav
      (A|AW|L|LW|N|R|RW) --
      -- N --          -- Default: show the whole list --
   multmem
      (single|list|corlist)
      list
   emptyanc
      (error|noterror)
      error
   HyNames
      CDATA
      "anchrole occrole"
>

<!ELEMENT
   nmsploc        -- Name-space location address --
                  -- Clause: 7.9.3 --
                  -- Addresses objects by name in a name-space --
                  -- Constraint: location source must be an object that
                     exhibits a named node list property or whose
                     additional property source exhibits a named node
                     list property. --
   - O
   (#PCDATA)      -- Lextype: (word|literal)* --
>
<!ATTLIST
   nmsploc

   HyTime
       NAME
       #FIXED
       nmsploc
   namespc        -- Name-space --
                  -- Name-space property of location source from which
                     nodes are selected. --
      NAME
      #REQUIRED
```

```
   locsrc         -- location source --
      CDATA       -- Reference --
                  -- Constraint: References to entities are references
                     to roots of primary groves constructed from the
                     entities. --
      #IMPLIED
   ordering       -- Is ordering of locations significant? --
      (noorder|ordered)
      ordered

   set            -- Make multiple a set by ignoring duplicates --
      (notset|set)
      notset
```

The local topic map dtd

```
<!-- This is a demo DTD for Topic Navigation Map documents -->

<?IS10744 ArcBase TopicMap>

<!NOTATION
   TopicMap       -- Topic Navigation Map Architecture --
                  -- A base architecture used in conformance with the
                     Architectural Form Definition Requirements of
                     International Standard ISO/IEC 10744. --

   PUBLIC "ISO/IEC 13250:1999//NOTATION AFDR ARCBASE
           Topic Maps//EN"
>
<!ATTLIST #NOTATION TopicMap
   ArcFormA       -- Architectural form attribute name --
      NAME        -- Constraint: must be unique among all attribute
                     names in DTD in which it is specified --
      TNM         -- Default: ArcName --

   ArcNamrA       -- Architectural attribute renamer attribute name --
      NAME        -- Constraint: must be unique among all attribute
                     names in DTD in which it is specified --
      TNMNames    -- Default: no renaming --

   ArcsuprA       -- Architecture suppressor attribute name --
      NAME        -- Constraint: must be unique among all attribute
                     names in DTD in which it is specified --
      sTNM        -- Default: no suppression --

   ArcIgnDA       -- Architecture ignore data attribute name --
      NAME        -- Constraint: must be unique among all attribute
                     names in DTD in which it is specified --
      TNMIgnD     -- Default: data is conditionally ignored --

   ArcDocF        -- Architecture document element form name --
      NAME        -- Constraint: name of element form in architecture
                     meta-DTD --
      topicmap    -- Default: ArcName --

   ArcDTD         -- Architecture meta-DTD entity --
      CDATA       -- Lextype: (ENTITY|PENTITY) --
      "%TopicMap" -- Default: entity whose name matches the notation name
                     after SGML name and entity folding is applied. --
                  -- Constraint: a name must be specified or the meta-DTD
                     entity name after folding must match notation name
                     after folding. --
```

```
   ArcBridF       -- Architecture bridge form name --
                  -- For elements with an ID and no ArcForm attribute
                     whose form isn't defaulted --
      NAME        -- Constraint: name of element form in architecture
                     meta-DTD --
      TNMBrid     -- Default: no defaulting --

   ArcAuto        -- Architecture automatic form mapping --
                  -- Automatic form mapping for identically named
                     element types and external data entity
                     notations --
      (ArcAuto|nArcAuto)
      nArcAuto
>

<!ENTITY % TopicMap PUBLIC "ISO/IEC 13250:1999//DTD AFDR Meta-DTD Topic Maps//EN">


<!ELEMENT mytopicmap - - (mytopic)* >
<!ATTLIST mytopicmap
   TNM     NAME  #FIXED topicmap
>

<!ELEMENT mytopic - - ((myname+, myoccur*) | myoccur+) >
<!ATTLIST mytopic
    TNM      NAME             topic
    id       ID               #REQUIRED
    public   ENTITY           #IMPLIED
    type     CDATA            #IMPLIED
>
<!ELEMENT
  myname            -- A name of a topic. --
  O O
  (myfullname, mydispname*, mysortname* )
                  -- If dispname or sortname are not specified,
                     applications use fullname.--
>
<!ATTLIST
  myname
   TNM        NAME    #FIXED  name
   scope          -- List of references to scoping topics that
                     collectively define the limited contexts within
                     which the name is applicable to the topic. --
      CDATA       -- Reference --
                  -- Reftype: topic+ --
      #IMPLIED    -- Default: scope is unconstrained. --
>
<!ELEMENT
  (myfullname | mydispname | mysortname)
   - O
  (#PCDATA)
>
<!ATTLIST myfullname
   TNM
      NAME
      fullname
>
<!ATTLIST mydispname
   TNM
      NAME
      dispname
>
<!ATTLIST mysortname
   TNM
      NAME
```

```
      sortname
>
<!ENTITY %
   loc            -- Location address forms --
   "mynmsploc"
>


<!ELEMENT
  myoccur           -- topic occurrence --
                 -- Clause: 5.2.3 --
  - O
  (%loc;)*
>
<!ATTLIST
  myoccur
   TNM
      NAME
      #FIXED
      occur
   occrole        -- Occurrence role name. --
      NAME
      #IMPLIED
                 -- Default: occurrence role name is GI of
                    element. --
   scope          -- List of references to scoping topics that
                    collectively define the limited contexts within
                    which the occurrences are applicable to the
                    topic.  Each referenced topic may be regarded as
                    describing a class of occurrence.  --
      CDATA       -- Reference --
                 -- Reftype: topic+ --
      #IMPLIED    -- Default: The topic which has a name (within the
                    scope of the OCCTYPE public topic) which is the
                    same as the value of the occrole attribute, or,
                    if no value is specified for the occrole
                    attribute, the generic identifier.  Even if no
                    such topic exists in the topic map document(s),
                    it is considered to exist by virtue of this
                    defaulting mechanism. --
   linktrav
      (A|EI|ER|ED|EN|EP|ERD|I|ID|D|N|P|R|RD)
      A
   -- listtrav
      (A|AW|L|LW|N|R|RW) --
      -- N --          -- Default: show the whole list --
   multmem
      (single|list|corlist)
      list
   emptyanc
      (error|noterror)
      error
>
<!ELEMENT
   mynmsploc      -- Name-space location address --
                  -- Clause: 7.9.3 --
                  -- Addresses objects by name in a name-space --
                  -- Constraint: location source must be an object that
                     exhibits a named node list property or whose
                     additional property source exhibits a named node
                     list property. --
   - O
   (#PCDATA)      -- Lextype: (word|literal)* --

>
<!ATTLIST
```

```
   mynmsploc      -- Name-space location address --
                  -- Clause: 7.9.3 --

   TNM
      NAME
      #FIXED
      nmsploc

   namespc        -- Name-space --
                  -- Name-space property of location source from which
                     nodes are selected. --
      NAME
      #REQUIRED

   locsrc         -- location source --
      CDATA       -- Reference --
                  -- Constraint: References to entities are references
                     to roots of primary groves constructed from the
                     entities. --
      #IMPLIED
   ordering       -- Is ordering of locations significant? --
      (noorder|ordered)
      ordered

   set            -- Make multiple a set by ignoring duplicates --
      (notset|set)
      notset
```

The document instance

```
<!DOCTYPE mytopicmap PUBLIC "-//Steve Newcomb//DTD A TNM DTD for demo only//EN" []>

<mytopicmap>
  <mytopic id="i8981">
    <myname scope="normal">
      <myfullname>New York</myfullname>
    </myname>
    <myoccur>
      <mynmsploc namespc="elements">idfudi</mynmsploc>
    </myoccur>
  </mytopic>
</mytopicmap>
```

The Airline Example

```
<?IS10744 ArcBase HyTime>
<!DOCTYPE topicmap [
<!NOTATION HyTime
   PUBLIC "ISO/IEC 10744:1997//NOTATION AFDR ARCBASE
           Hypermedia/Time-based Structuring Language (HyTime)//EN"
>
<!ATTLIST #NOTATION HyTime
   ArcFormA NAME     HyTime
   ArcNamrA NAME     HyNames
   ArcsuprA NAME     sHyTime
   ArcIgnDA NAME     HyIgnD
   ArcDocF  NAME     #FIXED HyDoc
   ArcDTD   CDATA    "HyTime"
   ArcQuant CDATA    #FIXED "NAMELEN 9"
```

```
   ArcDataF NAME     #FIXED HyBridN
   ArcBridF NAME     #FIXED HyBrid
   ArcAuto  (ArcAuto|nArcAuto) nArcAuto
   ArcOptSA NAMES    "base links locs"
   hyqcnt   NUMBER   32
   base     CDATA    "bos bosspec"
   locs     CDATA    "agrovdef bibloc dataloc datatok grovplan listloc
                      mixedloc multloc nameloc nmsploc pathloc
                      pgrovdef proploc queryloc referatt refloc
                      reftype relloc spanloc treecom treeloc treetype"
   links    CDATA    "varlink"
   exrefs   NAME     exrefs
   manyanch NUMBER   #IMPLIED
>
<!NOTATION AFDRMeta
   PUBLIC "ISO/IEC 10744//NOTATION AFDR Meta-DTD Notation//EN"
>
<!ENTITY HyTime
   PUBLIC "ISO/IEC 10744//DTD AFDR Meta-DTD
           Hypermedia/Time-based Structuring Language (HyTime)//EN"
   CDATA AFDRMeta
>

<!-- specific -->

<!ELEMENT topicmap - - (city+)>
<!ELEMENT city - - (name*, (streetmap|mention)* )>
<!ATTLIST city
   TNM   NAME  #FIXED "topic"
   id    ID    #IMPLIED
>
<!ELEMENT streetmap - - (#PCDATA) >
<!ATTLIST streetmap
   TNM   NAME  #FIXED  "occur"
>
<!ELEMENT mention - - (#PCDATA) >
<!ATTLIST mention
   TNM   NAME  #FIXED  "occur"
>
<!ELEMENT name - - (fullname)>
<!ATTLIST name
   TNM  NAME  #FIXED "name"
>
<!ELEMENT fullname - - (#PCDATA)>
<!ATTLIST fullname
   TNM  NAME  #FIXED "fullname"
>

]>

<topicmap>
<city id="c1">
 <name><fullname>New
York</fullname></name>
 <streetmap>id1</streetmap>
 <mention>id2 id3 id4</mention>
</city>
</topicmap>
```

In this example, New York is the name for a city. The unique identifier for the topic " `New York` " is `"c1` ". There are four occurrences of New York: one is a street map, and three are mentions. The elements "streetmap" and "mention" contain address of information objects that contain the objects serving as targets for this link. New York is the name of this topic.

The anchor specification of this link are described using specific elements. This version of the code imposes a DTD, in order to make sure what function each element plays regarding the TNM syntax.

Here is the DTD for this topic map:

```
<!ENTITY % occur "(mention | streetmap)" ><!ELEMENT city - - ( (name+, %occur;*) | %occur;+ ) >
<!ATTLIST city
   TNM       CDATA            #FIXED      "topic"
   id        ID               #REQUIRED
   HyTime    (varlink|HyBrid) varlink
   public    ENTITY           #IMPLIED
   type      CDATA            #FIXED      "city"
>
<!ELEMENT name - - (fullname, dispname*, sortname* ) >
<!ATTLIST name
   TNM       CDATA            #FIXED      "name"
   scope     CDATA            #IMPLIED
>
<!ELEMENT fullname - - (#PCDATA) >
<!ATTLIST fullname
   TNM       CDATA            #FIXED      "fullname"
>
<!ELEMENT dispname - - (#PCDATA) >
<!ATTLIST dispname
   TNM       CDATA            #FIXED      "dispname"
>
<!ELEMENT sortname - - (#PCDATA) >
<!ATTLIST sortname
   TNM       CDATA            #FIXED      "sortname"
>
<!ELEMENT mention - -  (#PCDATA) >
<!ATTLIST mention
   TNM       CDATA            #FIXED      "occur"
   HyTime    NAME             #FIXED      "anchspec"
   occrole   NAME             #FIXED      "mention"
   type      CDATA            #FIXED      "mention"
   linktrav
      (A|EI|ER|ED|EN|EP|ERD|I|ID|D|N|P|R|RD)
                                           "A"
   listtrav
      (A|AW|L|LW|N|R|RW)                   "N"
   multmem
      (single|list|corlist)                "list"
   emptyanc
      (error|noterror)
      error
   HyNames
      CDATA
      "anchrole occrole"
>
<!ELEMENT streetmap - -  (#PCDATA) >
<!ATTLIST streetmap
   TNM       CDATA            #FIXED      "occur"
   HyTime    NAME             #FIXED      "anchspec"
   occrole   NAME             #FIXED      "mention"
   type      CDATA            #FIXED      "mention"
   linktrav
      (A|EI|ER|ED|EN|EP|ERD|I|ID|D|N|P|R|RD)
                                           "A"
   listtrav
      (A|AW|L|LW|N|R|RW)                   "N"
   multmem
      (single|list|corlist)                "list"
   emptyanc
```

```
      (error|noterror)
      error
   HyNames
      CDATA
      "anchrole occrole"
```

Here is the same document, with no DTD needed.

```
<topic type="city" id="c1"> <name><fullname>New
York</fullname></name>
 <occur occrole="streetmap">id1</occur>
 <occur occrole="mention">id2 id3 id4</occur>
</topic>
```

Example 1. One topic

Code

```
<?IS10744:arch
   name="TNM"
   public-id="ISO/IEC 13250:1999//NOTATION AFDR ARCBASE
              Topic Maps//EN"
   dtd-public-id="ISO/IEC 13250//DTD AFDR Meta-DTD
                  Topic Maps//EN"
   dtd-system-id="c:\docs\topicmap\arch\tnm.dtd"
   form-att="TNM"
   renamer-att="TNMNames"
   suppressor-att="sTNM"
   ignore-data-att="TNMIgnD"
   doc-elem-form="topicmap"
   bridge-form="TNMBrid"
   auto="nArcAuto"
?>
<!DOCTYPE topicmap [

<!ELEMENT topicmap - - (city+)>
<!ELEMENT city - - (name*, (streetmap|mention)* )>
<!ATTLIST city
   TNM   NAME  #FIXED "topic"
   id    ID    #REQUIRED
>
<!ELEMENT streetmap - - (#PCDATA) >
<!ATTLIST streetmap
   TNM   NAME  #FIXED  "occur"
>
<!ELEMENT mention - - (#PCDATA) >
<!ATTLIST mention
   TNM   NAME  #FIXED  "occur"
>
<!ELEMENT name - - (fullname)>
<!ATTLIST name
   TNM  NAME  #FIXED "name"
>
<!ELEMENT fullname - - (#PCDATA)>
<!ATTLIST fullname
   TNM  NAME  #FIXED "fullname"
>
```

```
]>

<topicmap>
<city id="c1">
 <name><fullname>New York</fullname></name>
 <streetmap>id1</streetmap>
 <mention>id2 id3 id4</mention>
</city>
</topicmap>
```

Explanations

Architectural form declaration

This example starts with the architectural form declaration. It is a processing instruction, recognizable because it starts with "". Note that the ending question mark is only required in XML. It is not necessary in SGML. It is a good thing to include it anyway, since this guarantees compatibility between XML and SGML.

The processing instruction contains the following attributes:

name
Name of the topic. Here the topic has only one name, and this name only contains the `fullname` element.

dtd-public-id
Public identifier of the document type definition. File is found in the catalog.

dtd-system-id
File name contains the document type definition.

form-att

renamer-att

suppressor-att

ignore-data-att

doc-elem-form

bridge-auto

auto

Document type #### Topic "city" #### Occurrence "streetmap" #### Name and fullname ## Topic with one name

Code

```
<city id="c1">
  <name><fullname>New York</fullname></name>
</city>```##
Topic with one name and one occurrence

### Code
```

New York id1 ``` ### Explanations mention contains a pointer to an element that contains the address of the mention of New York.

It is possible that id1 instead contains the address of a group of elements. In other words, id1 can be the identifier of an address element, rather than the address of the information object itself.

Topic with two occurrences

Code

```
<city id="c1">
  <name><fullname>New York</fullname></name>
  <mention>id1</mention>
  <mention>id2</mention>
</city>```
###
Explanations
Here, there are two objects serving as occurrences for the topic
"New York". An alternative and equivalent syntax is :
<pre><mention>id1 id2</mention>```##
Topic with several occurrence roles

### Code
```

New York id1 id2 id3 ``` ### Explanations The occurrences are listed after the name of the topic.

An alternative notation for occurrences would be:

Topic with display name, sort name

Code

```
<city id="c1">
  <name>
    <fullname>New York</fullname>
    <dispname>New York</dispname>
    <sortname>NEWYORK</sortname>
  </name>
  <mention>id1</mention>
  <mention>id2</mention>
  <streetmap>id3</streetmap>
</city>```##
Topic with several names in different scopes

### Code
```

New York New York City Big Apple id1 id2 id3 ```## Topic with several names in different scopes, and display/sort names

Code

```
<city id="c1">
  <name scope="usual">
    <fullname>New York</fullname>
```

```
    <dispname>New York</dispname>
    <sortname>NEWYORK</sortname>
  </name>
  <name scope="formal">
    <fullname>New York City</fullname>
    <dispname>New York (City)</dispname>
    <sortname>NEWYORKCITY</sortname>
  </name>
  <name scope="casual">
    <fullname>Big Apple</fullname>
    <dispname>Big Apple</dispname>
    <sortname>BIGAPPLE</sortname>
  </name>
  <mention>id1</mention>
  <mention>id2</mention>
  <streetmap>id3</streetmap>
</city>```##
Topic having two names in the same scope

### Code
```

New York Nueva York id1 id2 id3 ```## Facet types and values

Code

```
<language>
  <english>n1 m1 m2 s1</english>
  <spanish>n2</spanish>
</language>```##
Introducing faceted occurrences

### Code
```

New York Nueva York id1 id2 id3 id4 n1 m1 m2 s1 n2 s2 ``` ### Explanations A streetmap of New York in Spanish has been introduced in the topic map. ## Two topics with different types

Code

Expressing types and scopes as topics

Code

Bibliography

SGML Biblio

- <cite>ISO 13250 CD (Topic Maps)</cite>

- <cite>ISO/IEC 10744:1992 (HyTime)</cite>

- <cite>ISO/IEC 10744:1997 (HyTime),</cite> 2 nd

edition, URL: http://www.ornl.gov/sgml/wg8/docs/n1920/

- Eliot Kimber, A Tutorial Introduction to SGML Architectures,
©; 1997, Isogen && E. Kimber,
www.isogen.com/papers

- <cite>HyTime User's Group Web site</cite> . URL:
www.hytime.org . Reference texts about HyTime.

- <cite>Practical Hypermedia: An Introduction to HyTime</cite>
. Kimber, William E.. Prentice-Hall Professional Technical
Reference. ISBN 0-13-309899-0. Draft available for review. Due to be
published mid 1998. URL: http://www.drmacro.com/bookrev

- ISO SC34 Web Site (was WG8 and WG4)

Bibliography

Glossary

Added Scoping Topics

Scopes are used to delimit the validity domain in which semantics is assigned to a set of information objects in a topic map. Scopes are declared by using topics, called *scoping topics* . They are usually explicitely created by the Topic Map designer. In addition, each topic map defines its own *name space* , enabling to trace the origin of all names used in the various constructs of the topic map. Therefore, even if the users do not explicitely indicate that scoping topics are being used, some of them are used anyway, by virtue of the fact that they are present in a given topic map. These scopes are added to the scopes declared by the topic map designers. The existence of these implicit supplementary scoping topics enable the merging *Merge* of various topic maps, since this feature is the one which enables users to identify from where a given scope comes from.

Added scoping topics allow name spaces to be used consistently over multiple topic maps.

The use of added scoping topics has nothing to do with the fact that scoping topics have been created or not by the designers of a given topic map. Even when ne scopes are used, there is anyway an added scoping topics that contains the name of the local topic map.

Added Scoping Topics Attribute

Added scoping topics can be specified **within** the TNM document whose scopes are affected, by adding this attribute on the document element. By doing so, topic map authors are declaring the *name space* that they consider to be used when refering to a given topic map.

Added Scoping Topics Data Attribute

Added scoping topics can be specified **outside** the TNM document whose scopes are affected, by adding a data attribute on the entity declaration of a given TNM document. This situation applies when the Topic Map that is referenced does not define a name space that is considered appropriate for merging.

Addressing Mechanism

An addressing mechanism is the way an address of an information object is expressed. The HyTime standard contains many addressing mechanisms. Because the topic map architecture is a client of the HyTime architecture, all addressing mechanisms can, in principle, be used. However, the applications are responsable to understand the addressing mechanisms. A given application is usually limited to a set of addressing mechanisms that can be used. Usually, an

application declared the list of formats that it is able to address, therefore giving to its users the information of the capabilities, and limits, of this application.

The XML-based version of Topic Maps uses the addressing mechanisms defined by the *X-Pointer* specification of XML.

Addressing Scheme
An addressing scheme is the concept on which an addressing mechanism is operating. HyTime defines a set of addressing schemes. Other addressing schemes are defined by proprietary applications. HyTime's Location Address Module has been defined in such a way that it embraces all existing addressing schemes. Applications based on given formats are able to understand a set of addressing schemes.

Anchor

Anchor Role

Architectural Attribute Renamer Name (ArcNamrA)
The attribute architectural attribute renamer name (`ArcNamrA`) specifies the name of an architecture control attribute (the `ArcNames` attribute) that allows an application to substitute its own names for architectural attribute names *(Source: HyTime)* .

Architectural Document Element

Architectural Form

Architectural Form Attribute Name (ArcFormA)
The attribute architectural form attribute name (`ArcFormA`) specifies the name of an architecture control attribute that the client DTD must define for every architectural element type and notation to identify its architectural form *(Source: HyTime)* .

Architectural support Declaration
Used to describe the way an architecture is used.

Architectural suppressor Attribute Name
The attribute architecture suppressor attribute name (`ArcsuprA`) specifies the name of an architecture control attribute (the `Arcsupr` attribute) that controls suppression of architectural processing *(Source: HyTime)* .

Architecture Automatic Form Mapping (ArcAuto)
The attribute architecture automatic form mapping (`ArcAuto`) indicates whether to map element types and external data entity notations to identically named architectural forms (`ArcAuto`) or not (`nArcAuto`).

Architecture Bridge Form Name (ArcBridF)
The attribute architecture bridge form name (`ArcBridF`) specifies the name of an architectural form (the "ArcBrid" form) that bridges between the architecture and a non-architectural element or notation. It is the default architectural form for elements that are architectural but do not have a defined architectural form. Such elements are usually included because they exhibit an ID attribute.

Architecture Control Attribute

Architecture Data Form Name (ArcDataF)
The attribute architecture data form name (`ArcDataF`) specifies the name of an architectural notation form (the"ArcData" form) that is used as the default for external data entities.

Architecture Document Element Form Name (ArcDocF)

The attribute architecture document element form name (`ArcDocF`) specifies the name of the document element form (the "ArcDoc" form).

Architecture Entity Declaration

Architecture Ignore Data Attribute Name (ArcIgnDA)

The attribute architecture ignore data attribute name (`ArcIgnDA`) specifies the name of an architecture control attribute (the "ArcIgnDA" attribute) that controls whether the data content of an element is treated as architectural. (*Source: HyTime*)

Architecture Meta-DTD Entity (ArcDTD)

The attribute architecture meta-DTD entity (`ArcDTD`) identifies the external entity that contains the meta-DTD to which the architectural document instance conforms. The value of the attribute is either the name of a parameter entity prefixed with the pero delimiter or the name of a general entity. It must be subject to the same SGML declaration as the client document, except that its quantity set could differ. The difference is specified by the attribute architecture quantity set (`ArcQuant`).

Architecture Options support Attribute Names (ArcOptSA)

The attribute architecture options support attribute names (`ArcOptSA`) specifies the names of one or more architecture support attributes. The support attributes list the names of parameter entities whose entity text should be declared as "INCLUDE" in the meta-DTD. The default architecture support attribute is architecture options (`ArcOpt`).

Architecture Quantity Set (ArcQuant)

A list of quantity and value pairs as found in the quantity set parameter of an SGML declaration. The values for the quantities named in the attribute value apply to the meta-DTD and architectural instance in place of those in the client document SGML declaration. If no value is specified, then the name of the meta-DTD entity is taken to be the name of the architectural notation after the application of SGML name and entity case folding rules.

Association

User-defined relationship between two or more topics (*Source: TNM*).

Association link

Hyperlink element describing an association between two or more topics. The anchor roles attribute express the semantics of the relationship (*Source: TNM*).

Association role

The role played by an anchor of an association (*Source: TNM*). The role is only a string. The association role type points to a topic that contains the information on the association role type.

Association type

Class of an association, described by a topic. It can be specified by the `type` attribute, which points to a topic. It it is not specified, the *generic identifier* of the *association link* is considered the association type. (*Source: TNM*). More precisely, the generic identifier is considered the name (within the ASSOCTYPE scope) of a topic that defines the association type. If no such topic exists, it is considered to be created by virtue of this defaulting mechanism.

Attribute

Attribute List

Base Module of HyTime *Base Module*

Module containing the core facilities of the standard, especially the facilities for *Document Management* document management.

Bounded Object Set

Bounded Object Set Control Data Attribute

Bounded Object Set Exception Specification (bosspec)

Bounded Object Set Level
Default BOS level used by data entities declared in hub document.

Bounded Object Set Path

Bounded Object Set Priority
Unconditionally specify the BOS priority of objects declared by the last entity named in each BOS path, to the BOS level specified by this bosspec's boslevel attribute.

Class

Context-Free Content (HyCFC)

Data Entity

Display Name

Document Element

Document Entity

Document Grove Plan

DTD
Document Type Definition. Description of the structure of a class of documents, as well as declaration of all elements and attributes (tags) used for this given class.

Element Form

Entity

Facet
Source: TNM

Facet Link
Source: TNM

Facet Type
Source: TNM

Facet Value
Source: TNM

Full Name

Generic Identifier

Grove

Grove Plan

Hub

Hub Document

Hyperlink Module of HyTime *Hyperlink*

Describes the facilities for connecting information objects together. HyTime hyperlinks emphasize on linking semantics while leaving the addressing facilities to the Location Address Module *Location Address Module* . The hyperlink module is divided into two main parts: contextual links and out-of-line links, also known as independent links. The latter distinguishes between link elements which are fixed and link elements which are variable *Varlink* .

Hyperlink syntax

HyTime

HyTime Document

HyTime Document Grove Plan

Include In Bounded Object Set (Inbos)

Unconditional include in, or exclude from, bounded object set. Unconditionally include or exclude objects declared by the last entity named in each BOS path, to the BOS level specified by this bosspec's boslevel attribute

Instance

Location Address Module of HyTime *Location Address Module*

Groups all the HyTime addresssing facilities achieving the declared goal of being able to address anything, anywhere, at any time. The addressing facilities are divided into three main parts: addressing by name, by coordinate position and by property.

Location of the Objects

Location Source

Maximum Bounded Object Set Level

Bounded level of HyTime bounded object set when document is a *hub* or *subhub* .

Meta-DTD

Set of architectural forms, i.e., templates for declaring SGML DTDs. A meta-DTD describes an architecture *Architecture* .

Merge

Name

Name Space

Name To Be Used As Sort Key

Notation

Occurrence Role

Public Topic

Pero Delimiter

Reference Location Address (refloc)

Reference Location Addresses Type (loctype)

Reference Location Source (rflocsrc)

Associates referential attribute with their location sources.

Reference Location Span (rflocspn)

Names pairs of referential attributes that address spans when both attributes are specified.

Referential Attribute

Scope

Scoping Topic

Some topics are used to qualify the scopes that are active in a given topic map. Scopes are used to limit the validity of assertions that are made as part of a topic map. In turn, scopes themselves can be considered as topics. Therefore, a topic map contains two types of topics: regular topics, that are used to qualify the information objects that happen to be described within a given topic map, and topics used to qualify scopes. These are called " *scoping topics* ".

SGML

SGML Architectural Form

Declarations for templates of element, attribute list, or entity declarations. Architectural forms are more flexible than corresponding declarations in the DTD. Actual conforming instances might be less constrained than what the meta-DTD expresses. Therefore, architectural forms are building blocks for templates used for DTDs.

Span

Subhub

TNM

Topic

Source: TNM

Topic Characteristic Assignment

Source: TNM

Topic Link

Source: TNM

Topic Map

Source: TNM

Topic Map Application

Topic Map Processing

Topic Name

Source: TNM

Topic Navigation Map

Source: TNM

Topic Occurrence

Source: TNM

Topic Type

Source: TNM

Unique Identifier

Variable Link

XML

The eXtensible Markup Language, defined by the World Wide Web Consortium. Version 1.0 of XML has been released in March 1998.

X-Pointer

Addressing mechanism defined as part of the XML specification. X-Pointers enables to address XML documents.