Automated Assembly of Custom Narratives from Modular Content using Semantic Representations of Real-world Domains and Audiences

Joshua Wulf, David Jorm, Matthew Casperson, and Lee Newson
jwulf,djorm,mcaspers,lnewson@redhat.com

Red Hat Engineering Content Services

Abstract. We present an approach to automatically assembling customized technical documents covering a specified area of interest, tailored to the needs of a specific audience, and with a meaningful narrative structure using semantically annotated modular units of information (topics), and ontologies that describe the structure of the real-world domain of interest. In this paper we explore the nature of narrative, and how an automated document assembler can produce coherent narrative using semantic representation. We introduce a Semantic Publishing system, named Skynet, that implements these ideas in the context of documenting commercial software products.
Keywords: semantic publication, automated assembly, customized narrative

1 Introduction

A utomated Document Assembly is the aggregation by a software agent of smaller units of information into a larger structure to meet the information requirements of a specific audience.

We define the larger structure as a *document*, which may be instantiated as a static linear document on paper (or as a pdf), or as a customized view of hyperlinked text. Narrowing the definition used by Andre et al. [AFQ], we attempt to formally define a document as "a collection of information targeted to an audience interest, constrained to include relevant information and exclude irrelevant information, and grouped and sequenced to match the audience's hierarchy of concern".

We distinguish technical documents – documentation that seeks to inform the reader about factual information – from other types of documents such as prose, or fiction, which fall outside our definition of a document (see Wright [Wri]).

The optimum structure for a document is a function of audience range of interest, existing audience knowledge, and audience hierarchy of concern. To produce an optimum document structure, an automated document assembler must have knowledge of these three aspects of the audience. Additionally it requires knowledge of the domain within the audience's interest, a collection of modular units of documentation that describe this domain, and knowledge of how these units relate to the domain and each other.

This allows an automated assembler to produce a customized document describing the range of the domain that matches the audience's range of interest, in a way that matches the audience's hierarchy of concern and level of knowledge. This can be done in the form of bespoke pdf documents or by modifying the dynamic presentation of a hyperlinked web of information.

We will look at each of these four areas in turn: Semantic representation of the domain; Modular Units of Content; Semantic Representation of an Audience; Automated Custom Assembly.

Before examining these four areas, however, we explain the theory of cognition that informs our implementation.

2 Theory of Cognition

We use a model of communication based on the Theory of Cognition proposed by van Dijk and Kintsch [DK]. They propose a categorization of communication elements that includes *textual representation* - words used to describe something - and a *situation model* - an internal mental model. The situation model represents some real-world domain, while the textual model represents a situation model in language.

The process of technical communication in this model is one of deconstructing the internal mental model possessed by a subject matter expert, marshaling the elements of that mental model into a verbal or written representation, streaming that representation to a receiver, demarshalling those verbal or written representations into mental elements in the mind of the receiver, and integrating those elements into the receiver's internal mental model.

All three of: the textual representation; the situational model; and a mapping between the two, must be available to the automated processor. We will first examine the situational model, which is made available to the automated processor as a *semantic representation of a real-world domain.*

3 Semantic Representation of a Real-World Domain

A situation model is an internal predictive mental model that is used to predict how objects in the real-world will act and react. The situation model encodes categories, membership, and relationship between elements of the world of experience [Gar] [Joh].

The structure of a document is itself semantic - the spatial and temporal relationships between textual units convey information about the relationships between the real-world elements that the textual units represent. Important information appears before less important information; things that occur or are encountered first are presented first; dependencies are presented before the things that depend on them. These kinds of decisions about the structure of a document are made by a human author using their own internal mental model. In our experience, in cases where a human author is missing or has an incomplete mental model of a domain, they must take recourse to subject matter experts for guidance on where to place information (for an illustrative example, see this discussion).

We define a *coherent narrative* as a semantic structure composed of comprehensible textual units. In our system textual units are authored by human authors, and their assembly into a meaningful (semantic) structure, or coherent narrative, is the role of the automated processor.

We conceptualize a situation model as an n-dimension hypercube which encodes a multiplicity of relationships along different dimensions of interest. This n-dimensional hypercube enables us to formally (and programatically) answer the question: "*Why is it that certain sentences should be "close" to each other in an instructional document ... ?*"[Hor] This Semantic Representation of a Real-World Domain (Domain Model) acts as a situation model for a automated document assembler that performs the role of human author/subject matter expert in deciding what information to include/exclude, and how to structure it in response to a given audience.

The Domain Model is implemented as categories and tags, which represent dimensions and points, respectively, in the n-dimensional hypercube of the situation model that the Domain Model simulates.

3.1 Ordered and Unordered Dimensions

Construction of a coherent narrative involves grouping and sequencing operations. Related information is grouped into sections and chapters, sequenced within that container, and the containers are themselves are grouped and sequenced.

We distinguish between *ordered* and *unordered* dimensions in the Domain Model, to encode the bases for these grouping and sequencing decisions.

Ordered dimensions are those dimensions whose points have an intrinsic sequential relationship that should be considered when constructing a narrative. Examples of such dimensions include "Lifecycle" (which is tied to dependency and also to time), "Temperature", and "Location" (for example: layers in a software stack).

Unordered dimensions are those dimensions whose points belong to the dimension, but do not have an intrinsic sequential relationship. Examples of such dimensions include: "End User Demographic", and "Name".

Unordered dimensions cannot be used as the basis for sequencing operations, and when information must be sequenced on the basis of a common unordered dimension the correct convention to use is "alphabetical ordering", a structure that semantically communicates: "*There is no meaning to this ordering*" [1].

[1] It is important to note for implementation purposes that alphabetical ordering is completely extrinsic to the semantic dimension, as it will change in the document output depending on the target language.

Table 1. Example Ordered Dimension in a Domain Model

Category	Tag	Tag Order
Lifecycle	Prerequisites	0
	Download	1
	Installation	2
	Configuration	3
	Deployment	4
	Shut down	5
	Redeployment	6
	Upgrade	7
	Removal	8

Table 2. Example Unordered Dimension in a Domain Model

Category	Tag	Tag Order
End User Concern	Application Development	null
	Server Administration	null
	Migration	null
	Troubleshooting	null

Tags belong to categories, and categories that encode ordered dimensions (*Category.IsOrdered*) make use of the *TagOrder* property of tags to encode the ordered nature of the dimension.

4 Modular Units of Content

In order to construct a document, an automated assembler requires a semantic representation of a situation model, and modular units of content to assemble.

The modular units must be sufficiently atomic to be meaningfully mapped to points within the n-dimensional hypercube of the Domain Model. A modular unit may be mapped to multiple points within the Domain Model, but all of the content in the unit must map to the same point(s). Otherwise, inclusion of the content in the output document based on its mapping to the Domain Model will result in the inclusion of content "in the wrong place".

To achieve this level of atomicity we have adopted use of the Darwin Information Typing Architecture (DITA) [DITA] topic types. The DITA Topic Types are based in part on the Information Mapping work of Horn [Hor+1]. Documents produced using formal division into Information Mapping units have been shown to be more effective than those produced with an ad-hoc information architecture [Hor+2].

The information in a DITA topic is constrained to a single subject, and a single information role. This level of atomicity makes them ideal candidates for mapping to a Domain Model through metadata tagging. We implement the modular units using the DITA categorizations of concept, task, and reference topics, but using the Docbook XML schema, to leverage our existing open-source Docbook publishing toolchain, Publican. It is the information typing aspect of DITA [IMI] that is of principal use and interest to us[1].

Constraining the content of a textual unit (topic) to a single *subject* means that they can be unambiguously mapped to points inside the n-dimensional hypercube of the Domain Model. This allows them to be reliably assembled according to the macrostructure of the document. Constraining them to a single *information role* means that they can be reliably assembled within that macrostructure, as concepts, references, and tasks play deterministic roles in the textual representation of an area of a situation model (something we will examine in due course). Some additional information may be required to assemble the units into a coherent narrative at this level. In addition to mapping topics to the Domain Model, topics can be mapped to each other. So a task can declare that a specific concept "is a dependency", or a reference can declare that it "illustrates" a specific task.

4.1 Natural Language Processing and Ontology Tagging

The content of the textual units (topics) is generated by human authors. The metadata tagging of the topics against the Domain Model and against each other is also performed by human authors. We have some rudimentary natural language processing tools that assist in this process. A conceptual vocabulary is generated based on the *Title* property of the topic. A scanner then examines the textual content of other topics, and suggests potential relationships based on the content, which can then be accepted or rejected by a human moderator. This is similar to the approach used by the BBC World Cup semantic publishing system [BBC].

We further examine the role that topic information role plays in assembling output when we examine Automated Custom Assembly.

5 Semantic Representation of an Audience

Generally speaking, an audience is a group of people with a shared interest and level of pre-existing knowledge [MS]. Audience is usually an approximation of a range, within which individual readers may completely or partially fit. The economics of document production dictate that a small number of documents be produced to serve a large number of people, and hence the idea of "audience" as a range. With automated assembly and electronic delivery, however, the economics of production of narrative change, and the problems associated with defining an

[1] although we also plan to support DITA XML encoded content in the future.

audience as generalized ranges [EL] [Ong] can be mitigated by using very specific definitions.

Different people are interested in different information, and they are interested in the same information in different ways.

Consider the following two cases:

1. An organization where the layers of a software stack are horizontally divided. In this case, one group is reponsible for the database component, and another group is responsible for the Operating System component.
2. An organization where the layers of a software stack are vertically divided. In this case, one group is responsible for installing both the Operating System and the Database, while another group is responsible for maintaining them.

In these two distinct cases the same information is needed by people in either organization, but it is needed in a different combination in each case.

Customized Narratives can be generated for each of these use cases. In the first case the narrative is generated by creating two documents. The information in the first document is constrained to topics tagged with the "Component:Database" tag. The information in the second document is constrained to topics tagged with the "Component:Operating System" tag.

In the second case, two documents are created. Both documents are constrained to ("Component: Database" OR "Component: Operating System"). The content of the documents is also grouped at the first level on these tags, resulting in two sections: "Database" and "Operating System". If the "Component" category is implemented as an ordered category based on the software layer, then the Operating System section will precede the Database section, otherwise they will be alphabetically ordered.

The first document is further constrained to information tagged "Lifecycle: Installation", and the second document is constrained to information (NOT tagged "Lifecycle: Installation").

Audiences can be further defined by linking an audience with a list of concepts that they can be expected to know. These concepts can then be elided from documents that are produced for this audience. Tasks can also be tagged with a tag from an ordered category "difficulty", and a threshold set for an audience, so that introductory and advanced guides can be produced.

When a formal definition of an audience is available to an automated processor, it can use this definition in conjunction with a semantic representation of a situation model and semantically-annotated modular units of content to assemble a custom narrative relevant to the audience's needs.

Because the production cost of this narrative assembly is so low, documents produced for an audience of one become economically feasible. The costs of narrative production move away from human authors (who can hardly be expected to write a different book for each reader), and move to processor cycles and a cognitive cost on readers, which we examine later.

6 Automated Custom Assembly

We use an algorithm to solve the general case of creating a coherent narrative structure from an arbitrary collection of textual units:

To automatically generate a semantic structure, we examine all of the topics that have been returned for a query, and assemble them into intermediate units based on topic types and any declared relationships between topics. We then examine the resulting aggregate units to determine if there exist meta data dimensions in which we can locate all of the units in the query.

1. If there is no meta data category for which all topics at this order of structure have a meta data tag, it's alphabetical ordering.
2. If all topics at this order of structure have a meta data tag from the same sequenced category, that category is a candidate for sequencing.
3. If all topics at this order of structure have a meta data tag from the same non-sequenced category, that category is a candidate for grouping.
4. If one grouping candidate exists and no sequencing candidates exist, group on that category and sequence alphabetically.
5. If one sequencing candidate exists and no grouping candidates exist, group and sequence on that category.
6. If more than one grouping or sequencing candidate exists, then follow the semantic rules for hierarchy of concern.

In addition to the interaction of audience range of interest and hierarchy of concern with the semantic representation of the situation model, the information type of the textual units (topics) influences the output structure of the document.

We use some basic patterns to structure the output, all other semantic considerations being equal, based on topic type. The basic pattern we use is "Concept, Task, Reference". Relevant or dependent concepts precede a Task, which is followed by additional reference material, including any example that illustrates its use.

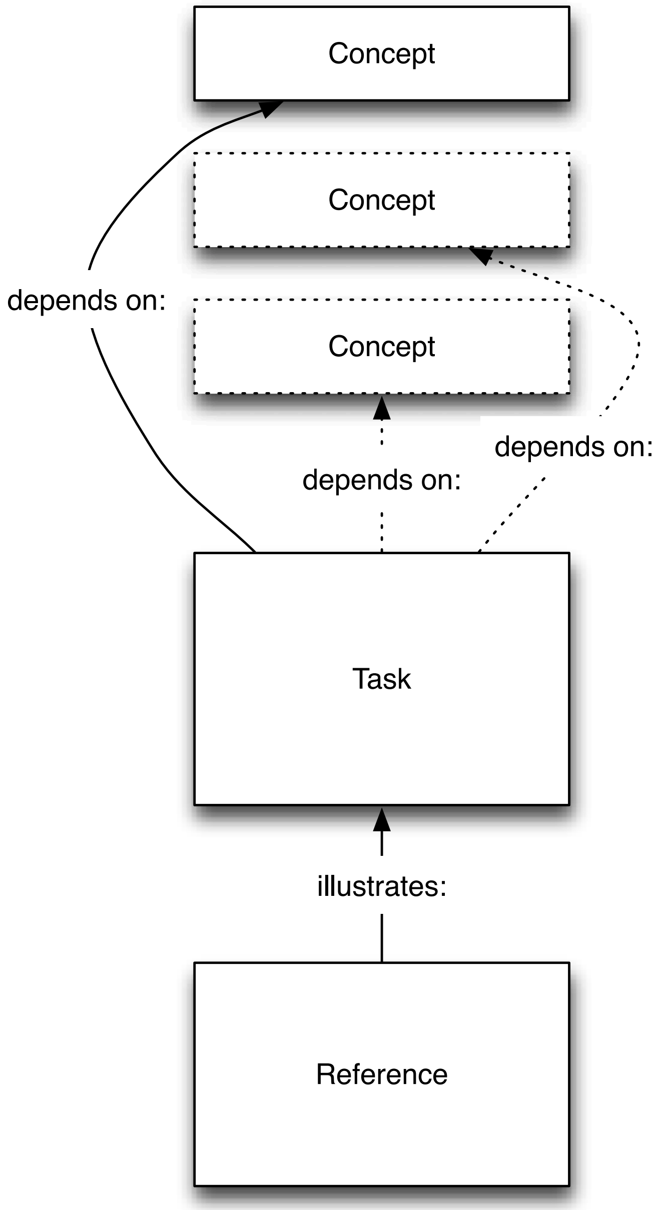
Figure 1 illustrates the output structure of a group of related topics, based on their topic type. The dotted lines represent concepts that may be elided or collapsed (in an html output) depending on the audience's level of knowledge, or the predicted relevance of the concept.

7 Current Status of Our System

Currently our semantic publishing system, known internally as Skynet and under heavy development, is implemented using as a JBoss Seam application, with a MySQL database to store the topics, and a topics-to-tags, tags-to-categories database schema to implement extensible meta data. Our processing engine is implemented as a combination of procedural code and rules using JBoss Drools.

The system is implemented as a platform, and has a REST API interface that allows it to be easily integrated and extended. One of the first extensions that we've developed for it is a content specification processor that allows an arbitrary

Fig. 1. An assembly pattern for textual content based on topic type



topic map to be passed to the system and returned as a Docbook book. This allows external semantic processors to generate their own output structures, and request the platform to build it from the content in the repository. This open and extensible design allows us to innovate and extend outside of the core code of the project.

The system is under development as an open source project on Sourceforge, and the current implementation is running on the Red Hat internal network and being used to develop the product documentation for the upcoming release of JBoss Enterprise Application Platform 6.

The documentation for JBoss Enterprise Application Platform 6 is written as modular content units (topics), and tagged against a semantic representation. The final output documentation is generated by an automated processor that locates the modular content units in an aggregate structure using the semantic representation to generate the document structure. In this sense it functions in much the same way as the BBC World Cup semantic publishing system [BBC] — the automated processor handles the publishing into the larger structure, while the human author is concerned with writing the content, and accurately describing it in terms of the semantic representation.

The document structure in Figure 2, for the JBoss Enterprise Application Platform 6 documentation, is generated by an automated processor using our semantic representation and semantically-annotated textual units. Information is constrained to the area of the hypercube containing "JBEAP 6" tagged topics. It is then grouped at the first level on the "Technology Component" dimension, which is an unordered dimension, hence the alphabetical ordering. At the second level of structure it is grouped on "End-User Concern", which is an ordered dimension based on lifecycle.

This is shown here as a tree structure, but it could also be instantiated as the table of contents of pdf output.

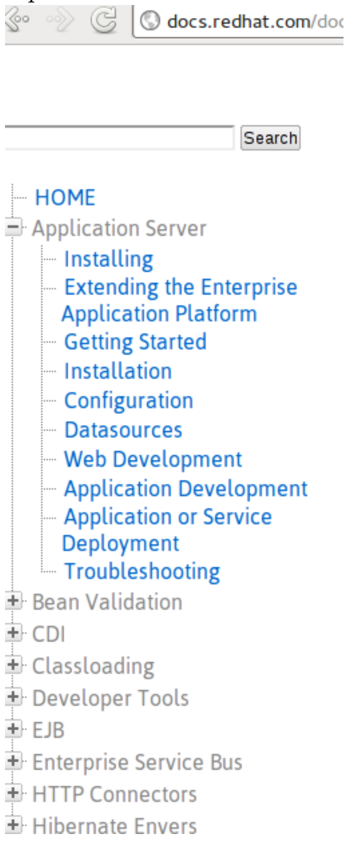
8 Challenges and Opportunities

At this point in time we are able to generate multiple output structures for various audience definitions. We have the data in place to allow the generation of Active Documents [DGK], however our production infrastructure currently serves over 2TB of documentation to users each month, so introducing dynamic content generation on our public-facing website represents a scalability and security challenge.

Our next plan is to make our dataset available to the public, possibly through a web service end-point, with our semantic representation available as RDF data. This will allow users to design their own documentation. In this case, users will be able to express their interest, either at high level or as a specific, complex query, and we will return a pdf file.

A significant challenge is encapsulating the complexity of the system. Users are not used to defining the content and architecture of documents. Making

Fig. 2. Output structure created by an automated processor using a semantic representation of a real-world domain and semantically-annotated textual units



the power of the system available to users without overwhelming them with its complexity is the greatest challenge and opportunity.

We are investigating three avenues to work towards this: natural language question answering; progressive customization of view; ant colony optimization of the semantic representation.

8.1 Natural Language Question Answering

Users are not accustomed to designing books, or articulating the exact nature of their domain of interest in a wide sense. They are, however, used to searching for information relating to a specific query that they have. With a semantic representation and semantically-annotated textual units, we are now in a position to investigate generating customised answers to queries from units of documentation. Rather than attempting to build a complete book, and requiring a user to specify an entire book, we would attempt to assemble the pieces to answer a specific query. Some disambiguation questions may be necessary to derive the exact area of interest, and the user's preferred format (hierarchy of concern), and then we can produce a small document to answer their query.

8.2 Progressive Customization of View

Rather than requiring users to define a view of the information, we can present users with a default view of the information (as we do now with JBoss Enterprise Application Platform 6 documentation). However, over time we can customize that view based on the user's behavior. We can infer things about the user based on their interaction with the material. When a user clicks on a search result, we know something about the result that they have clicked on. If we detect a preference to one area of the hypercube, we can weight search results to favor that area. We can establish *weak assumptions* about the user based on this kind of behavior. If we introduce the ability for the user to provide us with qualitative feedback, such as a "Like" or "this is what I was looking for" button, then we can also create *strong assumptions* about the user's preferences and further weight customizations.

Progressive customization of view is less taxing on users, although it may be more taxing on hardware requirements. Offline static rendering in response to a specific user query pushes the burden more to the user's side, and will be the first approach for the early adopters.

8.3 Ant Colony Optimization of the Domain Model

Ant Colony Optimization [DD] is a meta-heuristic optimization method, that simulates the behavior of an ant colony to approximate an optimal solution. It relies on many agents, in this case users, to iteratively explore the solution space and approximate a global optimum.

The Domain Model allows us to formally state "*Why is it that certain sentences should be "close" to each other in an instructional document ... ?*" [Hor].

However, there may be dimensions of interest to users that are not captured in our Domain Model, or are incorrectly encoded. If the Domain Model captures the situation model accurately, then rotation of the hypercube should allow two points in the model to come into proximity with each other. This means that information that is required sequentially by the user will be presented sequentially when the user's axis of interest is used to orient the hypercube. If we find that users consistently search for and then "like" two topics within a defined temporal period, and that these two topics are not available in proximity in a rotation of the hypercube, then it is an indication that there is something missing from the Domain Model, and this can be examined.

References

AFQ. Andre, J., Furuta, R.K., Quint, V: "Structured Documents", Cambridge University Press, 1989, pp. 3.

BBC. Rayfield, J., "BBC World Cup 2010 dynamic semantic publishing", BBC Internet Blog, 2010, http://www.bbc.co.uk/blogs/bbcinternet/2010/07/bbc_world_cup_2010_dynamic_sem.html.

DD. Dorigo, M., Di Caro, G. The Ant Colony Meta-Heuristic, IRIDIA Universit Libre de Bruxelles, 1999

DGK. David, C., Ginev, D, Kohlhase, M., Matican, B. Mirea, S., A Framework for Semantic Publishing of Modular Content Objects, In: *Proceedings of the 1st Workshop on Semantic Publishing 2011*, 2011, http://ceur-ws.org/Vol-721/paper-03.pdf.

DITA. OASIS Darwin Information Typing Architecture (DITA) Version 1.2 Specification, OASIS Standard, 2010, http://docs.oasis-open.org/dita/v1.2/spec/DITA1.2-spec.html.

DK. van Dijk, T. A., Kintsch, W.: Strategies of Discourse Comprehension, New York Academic Press, 1983, pp 305.

EL. Ede, L., Lunsford, A. "Audience Addressed/Audience Invoked: The Role of Audience in Composition Theory and Pedagogy", In: *College Composition and Communication*, Vol. 35, No. 2, National Council of Teachers of English, 1984.

Gar. Garnham, A.: "Mental Models As Representations of Discourse and Text", Ellis Horwood Ltd, 1988.

Hor. Horn, R.E., "Structured Writing as a Paradigm", N. J., Educational Technology Publications, 1998.

Hor+1. Horn, R.E., "Information Mapping", In *Training in Business and Industry*, Vol. 11, No.3, March 1974

Hor+2. Horn, R.E., "How High Can It Fly? Examining the Evidence on Information Mapping's Method of High Performance Communication", The Lexington Institute, 1992.

IMI. Information Mapping Institute Whitepaper, "Information Mapping©and DITA: Two Worlds, One Solution", Information Mapping Institute, 2011, http://www.informationmapping.com/us/resources/whitepapers/261-information-mapping-and-dita

Joh. Johnson-Laird, P.N.: "Mental Models: Towards a Cognitive Science of Language, Inference, and Consciousness", Cambridge: Cambridge University Press 1983.

MS. Mathes, J.C., Stevenson, Dwight W."Designing Technical Reports: Writing for Audiences in Organizations", The Bobbs-Merrill Company, Inc., 1976.

Ong. Ong, W.J., “The Writer’s Audience is Always a Fiction”, In: *PMLA*, Vol. 90, No. 1, Modern Language Association, 1975

Wri. Wright, P.: “Writing Technical Information”. In: *Review of Research in Education* Vol.14, (1987) pp.327-385 American Educational Research Association.