CHASID: High-level Authoring with consistency support for more efficient reading

Felix H. Gatzemeier, Oliver Meyer
Lehrstuhl für Informatik III, RWTH Aachen
E-Mail: mailto:fxg@i3.Informatik.RWTH-Aachen.DE
mailto:omeyer@i3.Informatik.RWTH-Aachen.DE
Project URL: http://www-i3.informatik.rwth-aachen.de/research/dfgspp

Summary

In the CHASID project (Consistent High-level Authoring and Studying with Integrated Documents), we create an authoring environment to leverage the semantical structure of documents to enable authors to produce better, richer documents faster. These can be handled and delivered by publishers more efficiently. We also support readers with tools to take full advantage of these enhanced documents.

Motivation and Goals

Knowledge structure visions

In the midst of this century, Bush (2) envisioned a knowledge system that stored and connected an uncountable number of informational bits. It would be an extension of his mind, freeing him from the limitations of his memory. Using it would blend naturally into his daily work. This vision has been the guiding light of many projects and is also cited as the root of the world-wide-web. The web does indeed contain a tremendous amount of information that is in some ways linked and searchable, but it is not the extension of the mind Bush predicted: inconsistencies, broken links and lack of deeper structure leave the user with information islands.

Deficit at the source of document creation.

One major reason for this unconnectedness is the high cost associated with preparing information in the required quality. Too much effort for too little use is needed in marking up documents. The same deficit affects conventional publishing, too: semantically marked-up documents could be tailored for specific reader needs, but preparing them is too expensive.

Authoring Support

The first aim of the project is a new kind of authoring support system. It does not only capture and arrange text, images and other media data, but helps the author to structure it. It takes into account that the author is starting out with a number of ideas to be expressed and ends up with a fixed product for distribution, similar to (6). It respects the author's choice of authoring system and produces documents in standardized notations.

Restructuring support

Writing a document is rarely finished in the first pass. After an initial sketch, parts are spelled out in detail, others are left open. Previously overlooked connections between topics surface and require attention. In discussing orthogonal dimensions, the primary dimension may change: starting with features of A and B followed by benefits of A and B, it may change to A's features and benefits, then B's. Reading may reveal weaknesses in the argumentation line, necessitating major restructuring. It is the purpose of this environment to help the author maintain semantical consistency across such operations.

Parameterization

The document may be adopted to different reader groups (engineers/managers), for different purposes (textbook/dictionary/slides for a talk) or to different media (printed book/hypertext in HTML). The rich semantical structure can be used to keep parts of the document together that are related. Common parts need to be written only once, changes in one variant are propagated into dependent variants.

Reader benefit

The resulting documents can be queried by the reader not only for keywords, but also for structures. Browsers can offer multiple views on the document through the structural information.

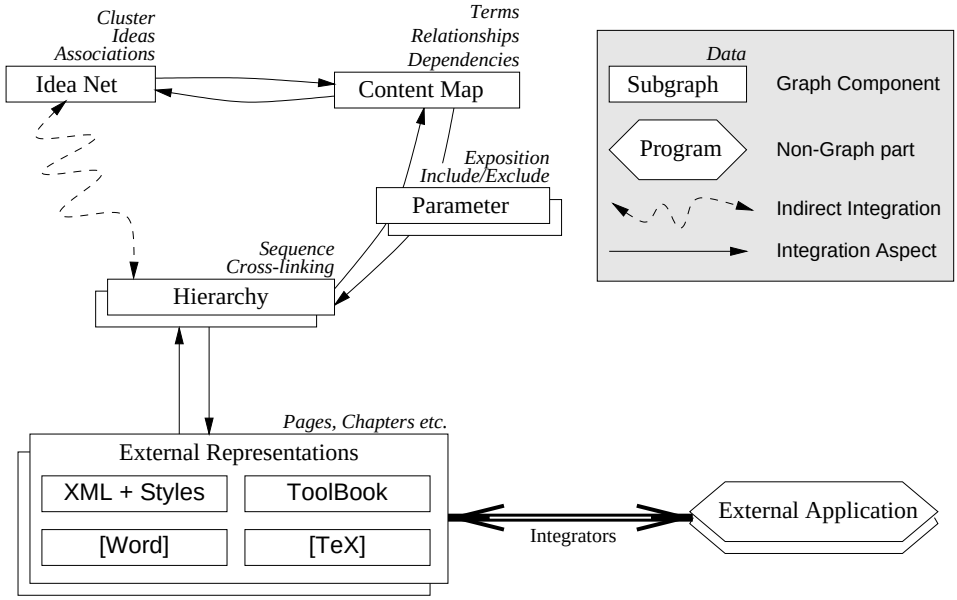

Figure 1: Data structures and integration

Approach

Writing process: three views

While all the above items are kept in perspective, current work focuses on operations to organize the inner structure of documents. A prototype of a semantically aware authoring tool is developed as an extension of conventional authoring environments. It offers three highly integrated views on the document:

Idea Net: Used to gather and cluster ideas that come to the author's mind in the context of the document. There is only little formal structure here. Ideas can be clustered; clusters may be collapsed for a broader perspective.

Content Map: Used to organize the content actually discussed in the document. Only a subset of the ideas will be relevant enough to be organized here. Content is classified into definitions, terms, arguments, examples, assignments etc. Relationships between content items are also classified: instance-of, example-for, required-for etc.

Hierarchy: An ordered tree of divisions (like chapters and sections) finally discussing the content. This subgraph connects to conventional authoring environments, which frequently do offer similar views. The connections are preserved during editing and are traversable both ways.

Authoring scenarios: Waterfall

Outline

The views are provided to allow the author to focus on specific aspects of his work, not to force him to adhere to one chosen writing process. • He may, as suggested by the order above, first collect a number of ideas, derive classified content aspects of the ideas and interconnect them to finally order them into a hierarchy, which gives him a document outline to fill with text and other media. But two-way integration allows for arbitrary choices of working styles: • If the author habitually first designs an outline of the document, with headlines and short reminder annotations, the semantics environment monitors editing operations of these well-known structures and derives its hierarchy graph from that. Headlines are kept as tentative content items, order is noted as probably-required relationships. Cross-references are likewise parsed and stored. Now, the author has a graphical view of his document and the relationships.

The graphical display in the Content Map suffices to give an overview of the connectedness between items. A flexible mapping from Hierarchy to concrete document allows to create different document variants. The author can also restructure the document in the Hierarchy as he is used to, and semantical data is used to mark relationships that broke up during the process.

Problems solved

Implementation

In the upper half of the figure the three views on the document can be seen. Changes in the Idea Net can be transfered to the Content Map and vice versa with direct tool support. Content Map and Hierarchy are also integrated in this way. Additional Parameters allow the creation of varying Hierarchies from one Content Map. The integration tools keep Idea Net and Hierarchy consistent, so that the author can check at any time whether all the ideas he wishes to mention are present in the document.

System structure

The lower half shows the subgraph structures especially adopted for the integration with existing authoring tools. Changes in the external document directly result in changes in these graph structures and are propagated to the Hierarchy and further.

The prototype is implemented using graph technology (4). The data structures are stored in a graph database and operations are specified using the high-level graph transformation language PROGRES (5).

Base Technology: Graphs

The fundamentals of handling text as a graph, deriving trees of variants in a document and the relationship between document structure and possible commands have been explored in a previous prototype. Current work on the CHASID prototype concentrates on developing the layered tool support and integration with common authoring environments, first with ToolBook.

State of implementation

References

[1] BEHLE, A., GATZEMEIER, F., AND MEYER, O. Graph technology for structured documents. In *Proceedings of SEKE '99* (June 1999), Knowledge Systems Institute, pp. 52–56.

[2] BUSH, V. As we may think. *Atlantic Monthly* (July 1945), 101–108.

[3] GATZEMEIER, F., AND MEYER, O. Improving the publication chain through high-level authoring support. In *Applications of Graph Transformation with Industrial Relevance (AGTIVE)* (1999), to appear in LNCS.

[4] NAGL, M., Ed. *Building Tightly Integrated Software Development Environments: The IPSEN Approach*, 1 ed. No. 1170 in Lecture Notes in Computer Science. Springer Verlag, Berlin, 1996.

[5] SCHÜRR, A. *Operationelles Spezifizieren mit programmierten Graphersetzungssystemen*. PhD thesis, RWTH Aachen, Deutscher Universitätsverlag, Wiesbaden, 1991.

[6] SMITH, J. B., WEISS, S. F., AND FERGUSON, G. J. A hypertext writing environment and its cognitive basis. In *ACM Hypertext'87 Proceedings* (1987), Invited Panel on Systems, pp. 195–214.