PROGRES und Fujaba wurden ebenfalls im Rahmen dieser Arbeit realisiert. Bei PROGRES wurden die Codegenerierung und das UPGRADE-Rahmenwerk angepasst. DRAGOS ersetzt nun das bisher verwendete GRAS vollständig. Seine Verwendung ist transparent und hat keinen Einfluss auf die Spezifikation oder die Werkzeugentwicklung. Die durch GRAS bedingten Einschränkungen sind weggefallen. Für Fujaba wurde ein spezielles Plugin entwickelt, das dessen Codegenerierung anpasst. Bis auf einige wenige Einschränkungen hat die Verwendung von DRAGOS keinen Einfluss auf die Spezifikation. Basierend auf den DRAGOS-Erweiterungen kann zukünftig die Funktionalität von Fujaba um komplexe Pfade, Nichtdeterminismus, etc. erweitert werden.

Die vorgestellten Konzepte wurden implementiert und validiert. DRAGOS stellt einen vollständigen Ersatz für seine Vorgänger dar. Durch seine offene Architektur und Erweiterbarkeit ist DRAGOS für zukünftige Anwendungen gewappnet. Die Probleme seiner Vorgänger werden somit langfristig vermieden.

Felix H. Gatzemeier: CHASID:A semantics-oriented authoring environment

Promotion: RWTH Aachen
Erstgutachter: Prof. Dr.-Ing. Manfred Nagl, RWTH Aachen
Zweitgutachter: Prof. Dr. Jörg Haake, FernUniversität Hagen
Datum der Prüfung: 10. Mai 2004
Veröffentlichung: 270 S., Shaker-Verlag, 2005

Kurzfassung:

In writing an informational document (such as a scientific article or a textbook), an author faces a number of complicated problems: not just creating the actual media (text, images, and so forth) with its intricacies of formulation, but also selecting the content to be presented and structuring it so that the document in its entirety is consistent and readable. The latter is complicated by re-iterations over the document, where the author reviews and edits it from different perspectives.

CHASID, the project described in this book, addresses these tasks of the author: (a) Planning a document, (b) Upholding plans, and (c) Avoiding common structural problems. To do so, it blends itself into the existing conventional authoring environment and maintains a semantical model of the document, which is connected to its hierarchical structure of chapters, sections etc. The functionality is organized according to a cognitive model of the authoring process.

The semantical model contains the topics of the document, together with their relations. It is thus also called the topic map here. The connection to the document's hierarchical structure is kept in exports and imports, indicating which divisions explain a topic, and which ones expect it to be known.

The additional functionality requires additional information which is not usually captured during authoring, namely the topic map and its connection. The author generally cannot be assumed to be willing to invest the effort of providing this information without seeing some direct benefit. So, the relationship between additional effort and kinds of possible functionality has also been considered.

Beginning with the least additional effort, requiring no additional information to be supplied by the author, tree transformations have been defined to support modifying the document hierarchy. For example, a division may be dissolved, promoting all of its children one level. This functionality requires the author only to find and understand the respective commands.

Addressing the overall planning problem, patterns have been introduced as a structured description of text types. A pattern has a name and consists of the problem to be solved (audience and content), a solution as a set of instructions, and a discussion pointing to alternatives and giving further advice. All these components are given in natural language. As a passive means of support (supplying documentation rather than functionality), they also do not require information from the author.

On the next level, schemata are provided, proven building blocks of documents spanning both the conventional document and the topic map. A schema has a name and consists of a short and a longer description and a part of a document with weighted components. When the author chooses to use a schema, the document part is merged into the existing document.

The new and the existing components participating in the schema instance are grouped in the document, and their weights in this context are recorded for later checks. These schema-based checks produce warnings if important or crucial schema components have been removed. To fix this, the author may connect another component from the document, or dissolve the schema instance. A third weight is used for optional components that may be removed without consequences.

This structure is still simple enough for interested casual users to understand: it is basically a cut-out of a document with some parts marked as more and some as less important. This allows schemata to be defined by an author, even for small substructures that just occur repeatedly within a chapter.

While instantiating schemata is the most convenient way to construct a topic map, operations for manual modeling are also available. On this level, the author invests the most effort and has to understand the types available in the topic map, but gets detailed control over the model. The author may insert topics and relations into the topic map, or remove them from it. With the topic map available, general checks can also detect faults in it, such as cyclic Part-Of relations.

A CHASID prototype has been implemented, connecting to ToolBook and Emacs as conventional authoring applications. The topic map and its connection to the conventional document are maintained in a graph database. All semantical operations have been implemented using the graph-based specification language PROGRES. User-defined schemata are stored as XML files conforming to a proprietary DTD.

The approach has been tested by translating advice from conventional writing guides into patterns and schemata. The results were generally satisfactory, but also revealed that more can actually be modeled than is commonly expressed in a guide.

In a side-track of the development, topic maps with only a synonym relation were regarded as a means to characterize and evaluate documents. Such models may be constructed by an attentive reader who does not have to be proficient in the subject area of the document, or they may even be derived automatically from documents written in sufficiently equipped markup languages.

Based on this, properties of topics relating to their order of exports and imports, being introduced only as a synonym, and others were defined. A formal concept lattice has then been constructed, regarding the topics and their properties as the objects and properties of a formal context. This lattice reveals characteristics of the document and can be used as a metric to spot trouble areas. For example, if many topics are imported before they are exported, an entire section may be misplaced.

The experiences from this project indicate that schemata can provide the basis for a semantical model far richer than what may be obtained by manual modeling. This improves creation as well as maintenance of documents.