STRUDEL: A Web-site Management System

Mary Fernandez, Daniela Florescu, Jaewoo Kang, Alon Levy, Dan Suciu
{mff,dana,kang,levy,suciu}@research.att.com
AT&T Labs
600 Mountain Ave., Murray Hill, NJ 07974

1 Introduction

The growth of the World-Wide Web has created a new kind of data management problem: building and maintaining Web sites. Building a Web site involves several tasks, such as choosing what information will be available at the site, organizing that information in individual pages or in graphs of linked pages, and specifying the visual presentation of pages in HTML. Creating and managing large sites is tedious, because a user often must perform these tasks simultaneously when creating HTML pages. The task of building and managing Web sites presents a unique opportunity for applying concepts from database management systems, such as the separation between the logical view of information and its storage and maintenance and the ability to restructure information via queries. Furthermore, recent research results on information integration [Ull97] and managing semi-structured data [ACM97, Abi97] can play a key role in managing Web sites. The goal of STRUDEL project is to extend and adapt these concepts to the problem of Web-site management.

Consider several tasks required of a Web-site manager. Site managers often want to manage a single repository of site data, but present different browsable "views" of the site based on criteria such as the type of user accessing the site, e.g., external or internal, expert or novice. Morever, a manager might want to modify the data repository by editing simple text files or by updating external databases, to reorganize the structure of the pages by manipulating graphs that represent the linked pages, or to design multiple presentations of a single page by editing HTML files or by using a WYSIWYG HTML generator. Currently, such operations on Web sites involve significant effort.

The key idea in the STRUDEL system is the *separation* of the logical view of information available at a Web site, the structure of that information in linked pages, and the graphical presentation of pages in HTML. Building a Web site using STRUDEL involves two steps. First, the Web-site manager defines independently the data that will be available at the site. Second, the Web-site manager decides how to structure and present that data. Intuitively, the structure of the Web site is defined as a *view* over the underlying data. STRUDEL allows users to manipulate the underlying data independently of where it is stored or how it is presented and to customize the Web site by creating different views of the underlying data.

Building STRUDEL poses several challenges. Defining the information available in the Web site may require integration of information from multiple sources (e.g., existing Web sites and various types of external databases). Defining sites as views over the integrated data requires a single query language whose results are Web sites, i.e., graphs of HTML pages. Data integration and view definition are complicated, because the data sources are not necessarily structured. Our solution in STRUDEL is to provide a single graph model in which all data sources are uniformly modeled and to provide a query language for both data integration and view definition.

Currently, there are two types of commercial products for Web site management. The first includes products such as Microsoft's Front Page and NetObjects Fusion, which are essentially WYSIWYG tools for creating HTML files. The second includes products, such as NetDynamics and Allaire's Cold Fusion, that automatically generate form templates for querying existing databases (such as Oracle and Informix) and for displaying the results of queries in HTML. Most database vendors provide similar tools for generating HTML interfaces.

In contrast to existing products, the goal of STRUDEL is *not* to provide a new HTML editor. In contrast to tools providing Web interfaces to existing databases, STRUDEL does not require the Web-site manager to store all the data in one given repository. Instead, STRUDEL provides a tool for integrating data from various sources. This is achieved by using a graph data model, which can accommodate data in existing models (e.g., relational or object-oriented) as well as legacy data or existing fragments of Web sites. We enable Web-site managers to view all the data in a Web site uniformly through our data model by using a mediated schema. As a result, we can query and manipulate the data to produce automatically various representations of the Web site.

2 System Description

Figure 1 depicts the architecture of the STRUDEL system. The main design principles of STRUDEL are:

Uniform graph data model. In every level of the STRUDEL system, data is viewed uniformly as a graph. At the bottom-most level, data is stored in STRUDEL's own graph data repository or in external sources (see Figure 1), which are viewed as graphs. STRUDEL's graph model is very similar to that of OEM [CGMH+94]; graphs contain *objects*, or named nodes, connected by edges labeled with attribute names. STRUDEL also provides *collections*, which are named sets of objects.

The *mediator* integrates multiple collections and objects into one *data graph.* STRUDEL's *site management engine* processes queries that compute *site graphs*, which are views of the data graph. A site graph is also a data graph, but whose nodes contain an attribute that specifies how to display in HTML the node's contents. Finally, the *HTML generator* materializes a site graph as a browsable graph of HTML pages.

Data integration. Defining the data graph requires integrating information from multiple sources. Our approach to data integration is similar in spirit to those developed in systems such as TSIMMIS [CGMH+94], HERMES [ACPS96], DISCO [TRV97] and Information Manifold [LRO96], SIMS [AKS96] and Occam [KW96], in the sense that STRUDEL defines a *virtual* loose schema and mappings between the contents of the data sources and the virtual schema. Our approach to object fusion is inspired by TSIMMIS [PAGM96]. The actual communication with the data sources is done through a set of *wrappers* [PGGMU95]. A STRUDEL wrapper maps an external source's data representation into STRUDEL collections and objects and translates STRUDEL queries into queries or operations understood by the source. Note that an external source can also be an existing Web site.

Single query and transformation language. An important advantage of using the same data model at all levels of the system is that we can use the *same* language for defining the data graph as views over the external sources (i.e., data integration) and for defining site graphs as views over the data graph (i.e., Web-site definition). An expression in STRUDEL's manipulation language has two parts: a query expression, which selects a subgraph of an existing graph, and a graph creation expression, which creates new nodes and links. A key feature of our language is the use of semantic oid's for creating new nodes. In addition, the selection component of the language supports *regular path expressions*, which specify how to match paths of arbitrary length in a data graph. Our language benefits from previous work in the area of querying unstructured data, graph data, and data with unknown schema [Woo88, BDHS96, MMM96, AQM+96, CCM96].

The following example STRUDEL query produces a site graph that contains `HomePage` nodes derived from objects in the `Manager` and `Person` collections in the data graph. The `FROM` clause specifies the collections from which the site graph will be created. The `WHERE` clause uses path expressions to match two kinds of paths in the data graph; the first expression matches any `Person` node `p` linked to a `Manager` node `m` by a `"Boss"` attribute; the second expression matches any `Person` node `p` linked to any other node by a `"Name"`, `"Address"`, or `"Email"` attribute.

```
FROM   Manager(m), Person(p)
WHERE  p "Boss" m
       p l q, l MATCHES "Name|Address|Email"

CREATE HomePage(p), HomePage(m)

LINK   HomePage(p) "Boss" HomePage(m)
       HomePage(p) l q
       HomePage(m) "Dept-Member" HomePage(p)
```

The `CREATE` clause specifies that new nodes should be created for each `p` and `m` node in the site graph; their unique identifiers are `HomePage(p)` and `HomePage(m)`. Finally, the `LINK` clause specifies how to link nodes in the site graph. Each person's home page points to his manager's home page and also preserves links to the person's original `"Name"`, `"Address"`, and `"Email"` objects. Managers' home pages are also extended to point to their employees' home pages.

Note that the definition of the site graph does not produce browsable, linked HTML pages. Instead, it produces an instance of a data graph that can be queried just like the original data graph. Moreover, the user can specify more than one HTML presentation for pages in the site graph. The HTML generator transforms a site graph into its final, browsable form.

Query processor for graph data. The primary focus of previous work on querying unstructured data has been the syntax and semantics of such languages. Only a few proposals (e.g., [CCM96, BDHS96, CM97] have addressed the problem of query optimization and evaluation. In the STRUDEL project, we have developed a general framework for query processing over multiple unstructured data sources.

As in relational systems, query optimization is done at multiple levels. Given a query, it is first translated into a relational expression over a set of finite automata. At this level, we apply optimizations such as factorization of common prefixes of regular expressions (in order to avoid multiple graph traversals when possible). The automaton representation is translated into

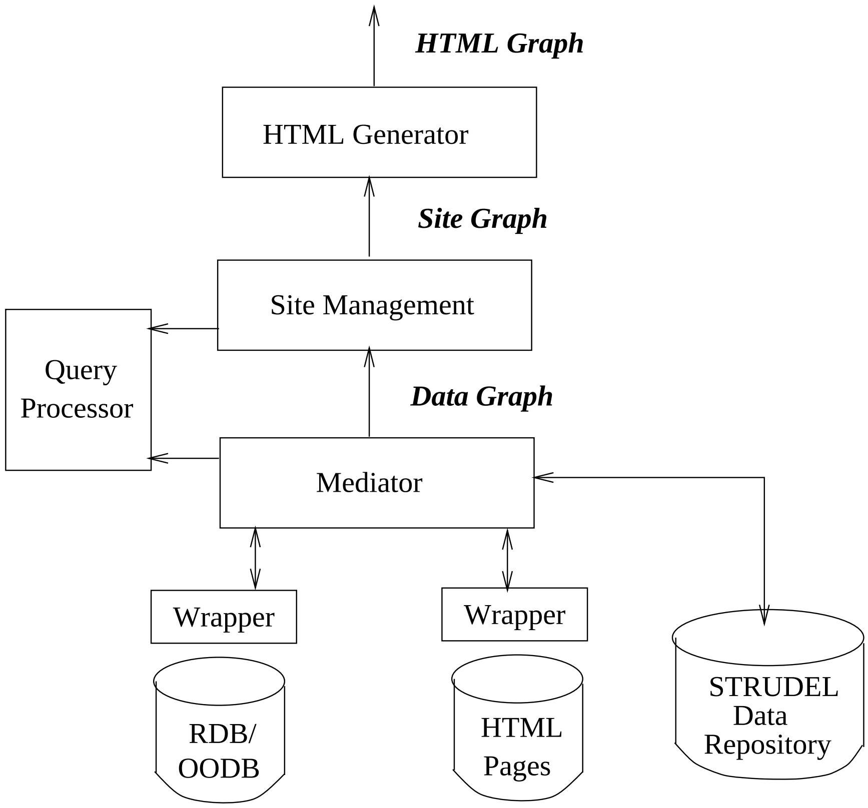

Figure 1: STRUDEL Architecture

the second-level representation, from which we produce query execution plans. A query execution plan is a set of atomic operations related by data flow arcs, possibly with cycles. For a given automata, the optimizer investigates a set of possible execution plans, using a dynamic-programming algorithm. In searching for the best query execution plan, the optimizer also considers the capabilities of the data sources (i.e., data access with limited patterns). The integration of the source capabilities into the cost estimation is a novel and important feature of our query processor.

3 Conclusions

Using STRUDEL to build and manage Web sites provides several benefits:

1. The separation between the logical view of the information underlying a Web site and its storage presents several advantages. First, the Web-site builder need not be concerned with issues such as updating and maintaining coherence of the HTML files containing the underlying data. Second, the logical view of the underlying information is more appropriate for the task of building a Web site and manipulation tasks such as restructuring, querying and analysis.

2. STRUDEL provides a mechanism that allows the automatic integration of multiple data sources when building the Web site. The system is not required to migrate all the data into one repository. Furthermore, since the data integration is a separate step in building the Web site (and is done as part of the process of building the conceptual model of the data), the resulting integrated view can be reused for creating different versions of the Web site.

3. The declarative nature of STRUDEL's view-definition language, the language that is used for specifying Web sites, makes it easy to restructure the site and specialize it for different classes of users.

4. Defining Web sites as queries makes it possible to enforce constraints on sites. There are two ways of doing this. First, some constraints are already implicit in the view definition. Second, additional constraints can be checked on the resulting site by posing specific queries.

5. By defining Web pages as queries, a user can specify concisely *sets* of similar pages and the corresponding links between them instead of designing each page individually.

6. When a Web site is constructed as a result of a query, one could build additional structures for *describing* the Web site (e.g., MCF [Guh97, BDFS97]). Such structures can be exploited by sophisticated

search engines or for building a site-specific search engine.

7. Because STRUDEL uses a graph data model, existing Web sites can be incorporated directly in the STRUDEL system.

Aside from the advantages for building Web sites, the concepts and techniques developed for STRUDEL are applicable in other domains. In particular, our query language and the architecture of our query processor can be used in other applications of graph data. Finally, STRUDEL employs novel techniques for information integration of semi-structured data.

References

[Abi97] Serge Abiteboul. Querying semi-structured data. In *Proceedings of the ICDT*, 1997.

[ACM97] Serge Abiteboul, Sophie Cluet, and Tova Milo. Correspondence and translation for heterogenous data. In *Proceedings of the ICDT*, 1997.

[ACPS96] S. Adali, K. Candan, Y. Papakonstantinou, and V.S. Subrahmanian. Query caching and optimization in distributed mediator systems. In *Proceedings of SIGMOD-96*, 1996.

[AKS96] Yigal Arens, Craig A. Knoblock, and Wei-Min Shen. Query reformulation for dynamic information integration. *International Journal on Intelligent and Cooperative Information Systems*, (6) 2/3:99–130, June 1996.

[AQM+96] Serge Abiteboul, Dallan Quass, Jason McHugh, Jennifer Widom, and Janet Wiener. The Lorel query language for semistructured data, 1996. Manuscript available from `http://www-db.stanford.edu/lore/`.

[BDFS97] Peter Buneman, Susan Davidson, Mary Fernandez, and Dan Suciu. Adding structure to unstructured data. In *Proceedings of ICDT–97*, 1997.

[BDHS96] Peter Buneman, Susan Davidson, Gerd Hillebrand, and Dan Suciu. A query language and optimization techniques for unstructured data. In *Proceedings of SIGMOD–96*, pages 505–516, 1996.

[CCM96] V. Christophides, S. Cluet, and G. Moerkotte. Evaluating queries with generalized path expressions. In *Proceedings of SIGMOD–96*, 1996.

[CGMH+94] Sudarshan Chawathe, Hector Garcia-Molina, Joachim Hammer, Kelly Ireland, Yannis Papakonstantinou, Jeffrey Ullman, and Jennifer Widom. The TSIMMIS project: Integration of heterogenous information sources. In proceedings of IPSJ, Tokyo, Japan, October 1994.

[CM97] Sophie Cluet and Guido Moerkotte. Query processing in the schemaless and semistructured context. Technical report, 1997.

[Guh97] R.V. Guha. Hotsauce mcf. http://mcf.research.apple.com/hs, 1997.

[KW96] Chung T. Kwok and Daniel S. Weld. Planning to gather information. In *Proceedings of the AAAI Thirteenth National Conference on Artificial Intelligence*, 1996.

[LRO96] Alon Y. Levy, Anand Rajaraman, and Joann J. Ordille. Querying heterogeneous information sources using source descriptions. In *Proceedings of the 22nd VLDB Conference, Bombay, India.*, 1996.

[MMM96] A. Mendelzon, G. Mihaila, and T. Milo. Querying the world wide web. In *Proceedings of the Fourth Conference on Parallel and Distributed Information Systems*, Miami, Florida, December 1996.

[PAGM96] Y. Papakonstantinou, S. Abiteboul, and H. Garcia-Molina. Object fusion in mediator systems. In *Proceedings of the 22nd VLDB Conference, Bombay, India.*, 1996.

[PGGMU95] Yannis Papakonstantinou, Ashish Gupta, Hector Garcia-Molina, and Jeffrey Ullman. A query translation scheme for rapid implementation of wrappers. In *In proceedings of the Conference on Deductive and Object Oriented Databases, DOOD-95*, 1995.

[TRV97] A. Tomasic, L. Raschid, and P. Valduriez. A data model and query processing techniques for scaling access to distributed heterogeneous databases in disco. *IEEE Transactions on Computers, special issue on Distributed Computing Systems*, 1997.

[Ull97] Jeffrey D. Ullman. Information integration using logical views. In *Proceedings of the International Conference on Database Theory*, 1997.

[Woo88] Peter T. Wood. *Queries on Graphs.* PhD thesis, University of Toronto, Toronto, Canada, M5S 1A1, December 1988. Available as University of Toronto Technical Report CSRI-223.