DBWiki: A Structured Wiki for Curated Data and Collaborative Data Management

Peter Buneman, James Cheney, Sam Lindley
School of Informatics
University of Edinburgh
Edinburgh, United Kingdom
{opb, jcheney}@inf.ed.ac.uk,
Sam.Lindley@ed.ac.uk

Heiko Müller
Tasmanian ICT Centre
CSIRO
Hobart, Australia
heiko.mueller@csiro.au

ABSTRACT

Wikis have proved enormously successful as a means to collaborate in the creation and publication of textual information. At the same time, a large number of curated databases have been developed through collaboration for the dissemination of structured data in specific domains, particularly bioinformatics. We demonstrate a general-purpose platform for collaborative data management, DBWIKI, designed to achieve the best of both worlds. Our system not only facilitates the collaborative creation of a database; it also provides features not usually provided by database technology such as versioning, provenance tracking, citability, and annotation. In our demonstration we will show how DBWIKI makes it easy to create, correct, discuss and query structured data, placing more power in the hands of users while managing tedious details of data curation automatically.

Categories and Subject Descriptors

H.3.5 [**Online information services**]: Web-based services

General Terms

Design

Keywords

databases; wikis; curation

1. INTRODUCTION

Curated databases are finding use in all branches of science and scholarship. Most curated databases are created and maintained in a collaborative effort by a dedicated group of people – the curators – who produce a definitive reference work for some subject area. A system that maintains curated databases faces several technical and usability challenges [10, 8]:

- Curated databases are collections of entries that predominantly follow a common structure (or schema). The database schema may need to change over time as the subject area evolves.
- Much curated data is copied and edited from existing sources. Since the value of curated databases lies in their quality and organization, knowing the origin of the curated data — its provenance — is particularly important.
- Previous versions of data need to be archived and easy to retrieve. The archiving system, furthermore, should support temporal queries over the history of data.
- In addition to the raw data, curated databases carry additional valuable annotations consisting of opinions of curators about the quality of data or suggested changes.
- Curators should receive credit for their contributions. Thus, the system needs to make data items citable and attributable to their contributors.
- Many data curation projects rely on their web presence to distinguish themselves from other projects; in fact, biological databases are *required* to have a web interface in order to be published in journals such as *Nucleic Acids Research*. Each database therefore needs a customizable web interface.

Both relational databases and wikis have strengths which make them attractive for use in collaboration. Wikis have proved enormously successful as a platform for collaborative creation and publication of textual information; they record detailed change histories and allow space for discussion. However, wikis primarily manage loosely structured hypertext and lack support for structured data, ad hoc queries, and fine-grained data provenance. Relational databases, on the other hand, have been optimized to handle structured data. Direct access to databases, however, largely remains the preserve of professional programmers and database administrators. Moreover, relational databases often lack native support for long-term versioning, annotation, and provenance. Relational database-backed applications can address these needs in principle, but in practice developing such systems requires hiring skilled (and expensive) programmers and database administrators, which is unrealistic for many small scientific database projects.

We believe that the needs of database curation projects could be met more reliably and cost-effectively by developing new general-purpose systems that combine the advantages of databases and wikis. We call such systems *Database Wikis*. Much of the basic research on curated databases needed to implement database wikis, such as archiving, citation, provenance, and annotation management, has already been conducted [7, 8, 11]. However, there is no single system that draws these techniques together.

1.1 Our Contributions

DBWIKI combines the ease of use and flexibility of a wiki with the robustness and scalability of a database; furthermore, DBWIKI implements previously-developed generic techniques for annotation, citation, provenance tracking, and versioning.

DBWIKI provides the ability to create, populate and browse curated databases using a standard web browser. Data entry and mod-

SIGMOD'11, June 12–16, 2011, Athens, Greece.
ACM 978-1-4503-0661-4/11/06.

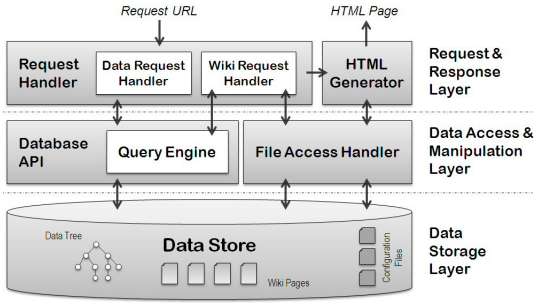

Figure 1: DBWiki System Architecture

ification is done either using system-generated web forms or by import from other data sources such as XML files. Each piece of information has a provenance record. All changes to the data and the database schema are logged in the database. The provenance and prior versions of each individual data item are browsable through the user interface. Moreover, each piece of information within the database can be annotated (including annotations themselves). Annotations are free-form text that allow curators to share and discuss their opinions, much like comments on blogs or forums; however, annotations can be attached to any part of the data, not just pages.

In addition to the database capabilities, DBWIKI includes a declarative "markdown" language for defining wiki pages. We extended a markdown syntax parser with syntax for embedded queries for viewing data. This extension enables querying and aggregating information from the curated database, *e.g., list the population of countries in Europe*. Similar proposals have been made for embedding SPARQL queries in other wiki extension projects such as Semantic MediaWiki [1]. DBWIKI's query capabilities, however, go beyond querying the current state of the data, as we also allow embedded queries over the history and the provenance of data, *e.g., list all changes made to the population of Greece*. Such queries are currently not supported by other wiki extension projects unless the history information is explicitly maintained in the data.

2. SYSTEM OVERVIEW

We implemented DBWIKI as a stand-alone Java application. The architecture, divided in three layers, is shown in Fig. 1. The bottom layer is responsible for storing the data and any additional information. The middle layer is responsible for querying and manipulating the data. The top layer of the architecture handles incoming request generates HTML pages in response. In parallel to the Java-based prototype presented here, we used a high-level Web programming language called Links [9] to develop another prototype, where we experimented with wiki-embedded query language design.

2.1 Data Model and Storage

Our Java-based prototype extends the XML archive management system XARCH [11]. The main extensions involve support for schema evolution, detailed provenance tracking, and annotation of data items. As with XARCH, DBWIKI is based on a hierarchical data model. We model the data using an ordered, deterministic data tree, equivalent to a simplified form of XML. Each node in the tree has a unique identifier. Each data tree is associated with a schema that describes the possible paths in the tree. DBWIKI currently supports a common set of data and schema modification operations including insert, delete, update, and rename. It is also easy to copy-and-paste nodes and subtrees within or among different curated databases. Each operation creates a new version of the database (efficiently using the archiving approach from XARCH). With each node we associate a timestamp that lists the database versions the node was present in. Based on the timestamp and information about the action that created each database version we derive provenance information for data nodes following the provenance model defined in [7]. Each node may also be associated with a list of annotations. Annotations do not create new versions.

We currently use a relational database back-end to store the data tree, annotations, and version information, *i.e.,* we shred the data tree, schema, and other metadata into relations. DBWIKI supports different database management systems using the Java JDBC interface. The relational database also contains the wiki page markup sources and configuration files used for web page layout (see below). Each of these files is also versioned.

2.2 Database Queries

With DBWIKI one can query the data tree and embed the results in wiki pages. Thus, DBWIKI's wiki pages are dynamic, combining hypertext with views of the structured data. Wiki page queries are translated to SQL queries against the relational data store. We currently support two different query formats. The first format uses the node identifier to retrieve a node (and its subtree) from the database. Queries may contain timestamp constraints to filter nodes in the subtree. The second query format is a special form of path expression, *i.e.,* sequences of node labels with optional constraints. Path expressions allow positional references as well as constraints on values of a node's children. For example, the query `/COUNTRY:2` returns the second country in the CIA World Factbook, a curated database of information about the countries of the world [2]. The query `/COUNTRY/CATEGORY/PROPERTY[NAME='Population']` returns the population for all the countries currently in the Factbook. We further allow constraints on the timestamps and provenance information of nodes, *e.g., the GDP of all countries that were updated by user admin in 2010.*

2.3 User interface

Users interact with DBWiki through a web browser, making requests encoded using URLs for either browsing or modifying the data, or for viewing or editing wiki pages. The URLs for wiki pages are similar to those in Wikipedia, *i.e.,* the page title is used as the page identifier. When browsing the data tree, we allow URLs similar to the query formats described in Section 2.2.

Once requested data has been retrieved, it is passed to the HTML generator that generates the response page. One of the design criteria of DBWIKI was to keep HTML generation separate from the rest of the system, hence highly customizable, as in typical wikis or content management systems. HTML generation is guided by three configuration files. The first file is an HTML template with placeholders for predefined user-interface components, *e.g.,* the "time-machine" interface for browsing the database history, HTML code for displaying and editing annotations, the history information, etc. Second, a cascading style-sheet (CSS) file is used to format the HTML output. The third configuration file is a layout definition that specifies how to map the tree-structured data to HTML pages, tables or lists. All files are editable through the user interface. Together, these configuration files give the user a great amount of flexibility in customizing the look-and-feel of the web pages.

3. DEMONSTRATION

We demonstrate the full capabilities of DBWIKI using data from several existing curated databases. Fig. 2 shows examples from our copy of the CIA World Factbook [2], including a wiki page with embedded query, the page markup, and the system-generated form

Figure 2: Wiki pages showing (from left to right) embedded query result, page source, and data update form.

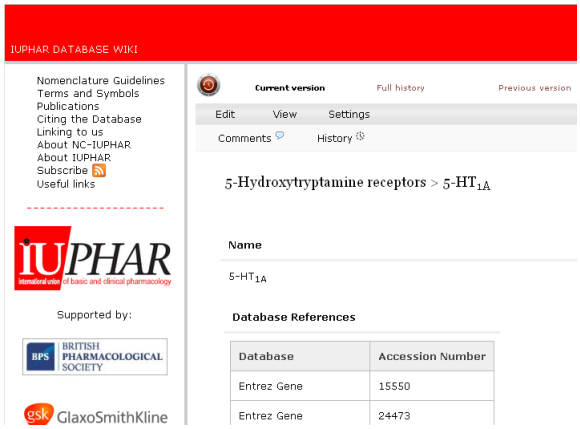

Figure 3: IUPHAR-DB web interface in DBWIKI

to update the data. Our initial setup includes a copy of the CIA World Factbook containing all changes to the database over the past three years, as well as copies of DBLP [3], IUPHAR-DB [5], and the Gene Ontology (GO) [4]. Using these databases we demonstrate (1) how to create a database from scratch, (2) collect data from other sources, (3) evolve the database structure, (4) query the history and provenance of data, (5) embed queries into wiki pages, and (6) customize the layout of web pages. In particular, we intend to show the following examples:

- We combine information from IUPHAR-DB, GO, and OMIM to create a new database on receptor involvement in human diseases. The example requires to copy, enhance, and restructure data from different sources.
- We extend the CIA World Factbook by importing information about capital cities from various sources like Wikipedia.
- Using the CIA World Factbook, we demonstrate query capabilities that include the history and provenance of data. For example, we answer questions like *Which countries have had at any point a GDP real growth rate over 5%*, or *Which countries were recently modified by adding new information*.
- We embed such queries into wiki pages and show how to browse the history and provenance of query results directly.
- We create a look-alike copy of the IUPHAR [5] database (see Fig. 3) – an important pharmacological database – from a given dump of the database by importing the data and editing configuration files.

These examples are only guidelines for our demonstrations and the audience is welcome to interact in the demonstration. In addition, we encourage the conference attendees to explore the capabilities of DBWIKI using a publicly available version. Thus, conference attendees will be able to start their own databases curation project, either from scratch or from a data file of their choice. Furthermore, everyone will be able to browse, modify, and query the existing databases we provide.

Acknowledgments. This work has been supported by EPSRC, Google and the University of Edinburgh IDEA Lab. Cheney is supported by a Royal Society University Research Fellowship. The DBWIKI system is available as an open-source project [6].

4. REFERENCES

[1] http://semantic-mediawiki.org.
[2] https://www.cia.gov/library/publications/the-world-factbook/index.html.
[3] http://www.informatik.uni-trier.de/~ley/db.
[4] http://www.geneontology.org.
[5] http://www.iuphar-db.org.
[6] http://code.google.com/p/database-wiki/
[7] P. Buneman, A. Chapman, and J. Cheney. Provenance management in curated databases. In *SIGMOD*, pages 539–550. ACM, 2006.
[8] P. Buneman, J. Cheney, W.-C. Tan, and S. Vansummeren. Curated databases. In *PODS*, pages 1–12. ACM, 2008.
[9] E. Cooper, S. Lindley, P. Wadler, and J. Yallop. Links: web programming without tiers. In *FMCO*, pages 266–296. Springer-Verlag, 2007.
[10] H. V. Jagadish, A. Chapman, A. Elkiss, M. Jayapandian, Y. Li, A. Nandi, and C. Yu. Making database systems usable. in *SIGMOD 2007*, pages 13-24. ACM, 2007.
[11] H. Müller, P. Buneman, and I. Koltsidas. XArch: archiving scientific and reference data. In *SIGMOD*, pages 1295–1298. ACM, 2008.