TreeTank, a native XML storage

Sebastian Graf

2009-03-27

Abstract

TreeTank is an easy-to-use framework which allows users to store, modify and retrieve tree-structured data. Modifications are encapsulated in different revisions which supports a navigation not only in the data-structure but also in the time-axis. Despite being an active research project, the software is covered with automatic test-cases and is used as a base for multiple student-projects and their thesis. TreeTank comes as a light-weight java program which allows easy handling in any java environment.

This report gives a short overview over the motivation, the technical foundation and projects based on with cited papers and an outlook in further extensions.

1 Motivation

The amount of data is increasing over the last year in a massive amount. Databases are nowadays not only a professional way to store massive amounts of data but also used in small applications. Nevertheless, the format of most databases is relational what means that the structure can be mapped in one or multiple tabulars. Meanwhile, new structures to handle data in a more efficient way entered the stage. E.g. tree-structured Data allow to access single data points without a sequential scan just because of the structure. That is why XML comes as a first choice of modern data formates. It is therefore not only seen as a data exchange format but also as a data storage format.

Even if there are multiple huge datasets which are stored in a XML-representation (Wikipedia, Swissprot, ...), there are only a few systems which are able to handle tree-structures even when it comes to multiple gigabyte of data. Since the structure of the tree is based on the flexible handling of data, this flexibility is bought with possible structural irregularities in the dataset. Since there are multiple approaches to map the tree-structure in a relational structure, these transformations scale bad when it comes to modifications in the tree since the mappings are based on the occurrence of the different in the nodes and has to be updated regarding insert/delete operations of nodes.

To overcome the scaling problem as well as the modification problem, TreeTank as a native XML-database was developed. N̈ativeḧere means, that the structure is stored as provided, as a tree and is not mapped in any tabular like schema. Additional to the possibility of offering modification in huge tree-structured datasets, revisioning was included in the update mechanism. Because information in XML is not only stored in the data itself but also in the relative occurrence of the nodes regarding to each other (e.g. ancestor–descendant relationships), the position of nodes plus their values can be directly be used as a base of well-known revision approaches.

To the best of our knowledge, there is no framework which support scaling regarding queries, modifications and revisions regarding tree-structured data. That is why we have multiple patents running on some ideas of TreeTank which are currently in the acceptance phase.

2 Technical Details

Java is chosen as the technical basis for TreeTank. Without any necessary external dependencies, TreeTank is able to run on every machine which has Java 6 SE installed. To provide trustable scientific results, TreeTank is covered with JUnit-Testcases which ensure a robust and (hopefully) bugfree framework. Build as a Maven Project, it offers a common software deployment process based on the well known spiral design in software-engineering. Treetank can therefore not only be seen as an active research tool but also as a robust peace of software which offers more stability and transparency than a prototype could ever offer.

TreeTank works on the backend with a RandomAccessFile which holds the data. In this file, the Tree-Structure is divided into the nodes which are combined on pages of the same size. Changes regarding the nodes results in a modification of only the related page. Therefore no complete revision of the whole tree-structure is necessary, only the modified nodes are stored. The nodes are distinguished regarding to their function in the tree. Since the data is represented over the nodes and the revisioning takes place on the nodes themselves, it is guaranteed that the revisioning is made upon the finest granularity level possible.

With the node level, TreeTank interacts similar to well known transaction concepts. Therefore, for each tree structure, multiple concurrent Read-Transactions can be build up to query the data in all available revisions. Additional to the Read-Transaction, currently only one Write-Transaction can be instantiated to make modifications on the structure this on the data. As discussed later, it is on the roadmap to provide multiple Write-Accesses bound to fixed substructures in the overall tree.

The storage itself is highly efficient. Since XML has the the reputation of being too verbose, TreeTank compressed the data in a way that it scales with the amount of information stored in the database. This is provided on the one hand by avoiding redundant storage of equal tagnames on the one hand and by an on-the-fly compression of the data itself. The compression just generates a constant overhead of time regarding the access of the data since the structure itself remains valid all the time.

Additional to the compression, check summing provides that the data and the tree-structure is valid over the time. This provides especially in heterogeneous environments, the certainty that no data is lost/corrupted when it comes to load-intensive modifications.

To provide efficient queries, XPath 2.0 is implemented in a way that support pipelining. That means that results of a query are not given out when the computation is finished but as soon as subsets of the queries are finished. The output is presented in a pipelined way which offers the possibility to handle even huge result sets and to given feedback to the user as fast as possible.

Encryption is provided for security reasons in TreeTank. Currently, the key must be known to readers and writers to access the structure. The encryption is provided in a straight-forward way. The data is accessed via an object–mapping and afterwards encrypted in the file using the provided key. Per instance, only one key is valid, but a multiuser encryption of relevant substructures is on the roadmap.

3 Implemented Projects

TreeTank has already been used in multiple projects. To give a short overview not only about the capabilities but also about possible use–cases, a short sketch up of existing projects is presented:

3.1 SVG–GIS Browser

[2] describes an approach which presents TreeTank as an backend for a web-based GIS. Despite of running a heavy weight application, the GUI is completely presented in SVG, an XML dialect which is commonly used to present vector-based graphics. Results show that when it comes to the use case to show XML-based gis-data, the speed is much better than using a PostGIS enabled Postgres–Database which generate SVG. [1].

3.2 Temporal REST

[3] presents an approach to exploit and modify XML–data with the help of the well-known REST–Interface and an extension for querying different versions. This offers the possibility to use TreeTank as an extra stand-alone data–service and not only the included part of any project.

3.3 Distributing XML–Databases

Since XML can be quite irregular regarding the structure, a distribution with focus on an equal load regarding queries is not as trivial as distributed relation data. [4] shows multiple approaches which are distributing the data to ensure equal loads and therefore regarding the query as well.

This is just a short overview of projects which were presented at conferences/workshops. All of these projects where based on Masterthesis which shows the capability of TreeTank as a storage platform.

4 Further Extensions

The work until now was focused on the backend of Treetank. Originally motivated to be a XML–Filesystem, the storage provides security, redundancy and efficiency. Nevertheless, TreeTank offers many possibilites to be accessed. To cultivate the idea of storing tree-structured and revisioned data, TreeTank will be further developed according to the following points.

4.1 Multiple Write Accesses in the Tree

Until now, TreeTank is only providing one write access in the complete tree structure at one time. Concurrent requests for writes can be made but are blocked since the concurrent existing write access is released. With concurrent possibilites for modification TreeTank will have the ability to increase performance and joy of use when if comes to multiple-user-accesses. Since we have an underlaying Tree-Structure, it is quite simple to provide concurrent accesses due to the different subtrees where an access can be bound to.

4.2 Increasing the supported interfaces

TreeTank is bound to multiple interfaces to provide it to a wider range of users. Currently, a DOM implementation is in development to provide easy access to the structure of the data. Besides the DOM, an JCR (Java Content Repository) integration is evaluated to increase the use of TreeTank for web-applications. With these two interfaces, common query and modification tools should be accessible soon.

4.3 Revisioned Index Structures

Index structures for revisioned data is a quite new field of research. In a masterthesis, a student developed an approach which allows revisioned access to indices as a base for revisioned data. This approach is just a prototype and has to be included into TreeTank to increase query performance.

4.4 Encryption of Nodes/Subtrees

Current status of encryption is the encryption of the whole tree-structure with only one provided key. To increase multi-user support, the encryption should not only take place on the complete data but on single nodes and subtrees as well. Providing one key per user, some subtrees are public and can be access by everyone while some other are protected for a closed circle of users. Sharing protected data to all possible users will be possible as well as encrpting public data which belongs to one user.

5 Conclusion

TreeTank promises a bright future as a database system. As a fully developed tool TreeTank will provide several novel features that are new to the world of databases, i.e. tree-structure as the base, check sums and encryption. Besides these features, TreeTank is plug-able tool to every java-based application without any additional installation, configuration and framework library. We believe that TreeTank will provide a new perspective not only to store data but also to the way they are accessed.

References

[1] Georgios Giannakaras. Design and evaluation of xml-based geographical visualization, 2009.

[2] Marc Kramis and Cedric Gabathuler. Streamlined workflow for large-scale interactive geographic visual analytics. In *Geospatial Visual Analytics Workshop, GIScience, 2008*, 2008.

[3] Marc Kramis and Georgios Giannakaras. Temporal rest - how to really exploit xml. In *IADIS International Conference WWW/Internet 2008*, 2008.

[4] Marc Kramis und Marcel Waldvogel Sebastian Graf. Distributing xml with focus on parallel evaluation. In *Sixth International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P 2008)*, 2008.