



Architecture and Design of 3-Way Merge Engine for Mobile Hosts

¹Mukesh Patel, ²Dr. K P Yadav, ³Dr. Swati Sharma

¹Computer Engineering, ²Computer Engineering, ³Electrical Engineering

¹NU Jodhpur, ²KCCITM Greater Noida, ³JNU Jodhpur

^{1,3}Jodhpur National University Jodhpur Rajasthan, ²Greater Noida, Jodhpur Rajasthan, INDIA.

Abstract: Keeping data up-to-date across a variety of devices and environments is becoming more and more important and challenging. The typical user of information technology no longer has access to single computer, but several, in the form of home and office workstations, personal digital assistants (PDAs) and mobile phones. Using several devices and environments has lead to the spreading out of the user's digital environment: his calendar may reside in the PDA as well as on the home and office PCs, his phone book in his mobile phone and office PC and his unfinished novel as an HTML document on the Internet as well as a word processor file on his home PC. Presenting in this paper the Architecture and Design of 3- Way Merge Engine for automatic integration of different versions of a tree data structure

Keywords: Architecture and Design, PDA, XML, 3-Way Merge Engine, Tree data Structure

I. Introduction:

Keeping data up-to-date across a variety of devices and environments is becoming more and more important. The typical user of information technology no longer has access to single computer, but several, in the form of home and office workstations, personal digital assistants (PDAs) and mobile phones. Using several devices and environments has lead to the spreading out of the user's digital environment: his calendar may reside in the PDA as well as on the home and office PCs, his phone book in his mobile phone and office PC and his unfinished novel as an HTML document on the Internet as well as a word processor file on his home PC. Having information in several places leads to the question of how to keep it all synchronized

II. Review of literatures:

The Concurrent Versions System (CVS) [1, 2] is a widely used document version control system. The architecture is client/server based with a central repository for the documents. The users check out local copies of documents from a repository, make changes, and commit the updated versions to the repository. Typically, check-outs in CVS are non-exclusive, meaning that multiple users may have a document checked out for editing simultaneously. As concurrent editing of a file is allowed, the need for merging changes sometimes arises. Typically, users A and B have checked out a file, they both modify it, and A commits the changes to the repository. Now, when B tries to commit his changes, CVS informs that the changes made by A need to be merged into his version before he may commit. CVS is able to automatically handle the merging quite well provided the document is a line based plain text file. Naturally, CVS cannot merge arbitrary binary files, and there is no particular support for merging structured data, except as plain text files.

Groove is also able to work in an online state and synchronize the shared space when the network becomes available again. This is accomplished by keeping a local history of all changes made and using a "relay service"[3, 4]

Lotus Notes by Lotus Development Corp. [Lot] and Groupwise by Novell [Nov] are two widespread examples of collaborative software, or "groupware", providing such functionalities as sharing and version control of documents, messaging and scheduling. Both are client/server solutions, meaning that the data is kept on centralized servers as much as possible [5, 6].

SyncML [7] is an XML-based data synchronization protocol pushed by the SyncML Consortium [SML], whose founding members include IBM, Lotus, Nokia, Ericsson and Motorola. The protocol is an attempt to standardize the way data synchronization messages are exchanged between devices. SyncML support is expected to start appearing in a wide range of handheld devices, such as PDAs and smart mobile phones. The basic architecture is a client/server configuration, where the server is constantly online, and the clients are off line for most of the time.

WebDAV (also known as just DAV), defined in [8] is an extension to HTTP that adds distributed authoring and versioning capabilities to the protocol. The purpose of WebDAV is to allow users to perform remote web authoring operations including creation, editing, copying, moving and deleting of web resources. A web resource is typically a file or collection of files identified by an HTTP URL. The computing model of WebDAV is client/server.

WebDAV includes locking of web resources. Both exclusive locks, which prevent users from simultaneously editing a resource and shared locks, which restricts simultaneous editing to those users sharing the lock, are supported. In the case of shared locking, a need for merge of the updated resources may arise. WebDAV does not define how the merge should be accomplished, though. [8]

III. Overview of differencing tools:

Differencing tools are relevant to the development of 3DM on two accounts: the first (and obvious) is that 3DM should be able to produce the difference, or delta, between two XML documents in order to save the amount of data that needs to be transmitted or stored. The second is that differences can be thought of as edit histories, and by combining these edit histories we can presumably perform a 3-way merge. In the following sections we will take a look at existing differencing tools.

1. UNIX diff:- The well-known UNIX differencing tool diff is capable of producing the line-by-line difference between two text files. Based on the algorithm by [9], it produces the minimum-length edit script to transform the first input file to the second using line insert and delete operations. Diff does not detect line moves; these are treated as delete/insert pairs. The tool is not directly applicable to structural differencing, as it is commonly believed that general structural differencing problems cannot be reduced to linear differencing problems [10].

2. TopBlend:- TopBlend [CDH00] is a differencing tool for HTML files, based on earlier work presented in [11]. TopBlend is able to show the differences between two HTML files in either as a side-by-side view or as combined HTML file. The tool is based on recursively applying the Jacobson-Vo [12] heaviest HCS (Heaviest Common Subsequence) algorithm to the different depth levels of the HTML parse trees. First, the children of the <BODY> tag of both files are compared. Using the HCS algorithm the children are aligned and corresponding tags in both files are identified. Then, taking the recursive step, the children of aligned tags are compared, just as the children of the <BODY> tag were compared in the first step.

3. XMLTreeDiff:- XMLTreeDiff by IBM Alphaworks [13] is a general-purpose differencing tool for XML files. The tool is based on the work by [14] and supports three basic node operations: updating, deleting and inserting a node. In addition, in order to increase the conciseness and readability of the diff, the tool has three aggregated operations: subtree prune (deleting an entire subtree), subtree graft (inserting a subtree) and subtree move (moving a subtree) [15]. The output of the tool is encoded in either of two XML-based update languages, both using the standard XPath syntax for identifying nodes in the source document.

4. LaDiff:- LaDiff [16] is a structural differencing tool for LATEX documents. In addition to handling updates, inserts and deletes, LaDiff also handles moves. The move operation is built into the LaDiff core algorithm, as opposed to the move operation of XMLTreeDiff. In practice this means that LaDiff identifies moves that XMLTreeDiff would treat as a pair of prune and graft operations. The program is more of a demonstration of the algorithms developed in [16] than a full-blown diff tool: only certain LATEX elements are supported. However, the main interest in this tool from the point of view of this thesis are the underlying algorithms.

IV. Overview of merging tools:

Merging tools are relevant to merge the two files of same data structure.

1. UNIX diff3:- The UNIX diff3 tool is a front end to the di_ tool which provides support for 3-way differencing and merging of text files. Assume we have a base document B, of which there are two branches B1 and B2. Diff3 accomplishes merging by either producing the diff $B \ominus B1$ and patching B2, or by patching B1 with the diff $B \ominus B2$. Being based on diff, the tool is inherently a plain text merge tool, and as such is not suitable for structural merging

2. BM Alpha works' XML Diff and Merge Tool (XMLDiff):- IBM's Alpha works have also developed a "XML Diff and Merge Tool" (XMLDiff) [17] independently of XMLTreeDiff. The tool compares two XML input files, and shows the differences. The user is then able to interactively merge the two versions by choosing either branch at the differing locations. The implementation details of the tool are not publicly available, but it appears (according to [18]) that the XML parse trees are compared one depth level at a time. Being a pairwise comparison tool XMLDiff cannot automatically perform merges. Its usefulness in the scope of 3DM is also limited due to the closed implementation.

In addition to the above mentioned general merge tools there are numerous application-specific merge tools, either integrated in the application or standalone. Examples include the "Compare Document..." tools in Microsoft Word and StarOffice and programs for synchronizing appointments between a PDA and your laptop. These tools are of little relevance to 3DM. The level of merging usually limited to line-by-line comparison, they are proprietary and application-specific and, as far as the author is aware, they do not exhibit any new thinking in merging algorithms.

V. The Architecture and Design of 3-Way Merge Engine

We are presenting the modular structure of the complex architecture for the 3-Way Merge Engine. The overall Architecture consists below four parts:

1. Input File reader or Parser
2. Tree Matcher
3. Three-way Merger
4. Tree Differentiator

Below Figure 1 illustrates the different modules and their internal relations. A brief description of each module described as below:

1. Input File reader or Parser: This is the important Module of the Engine where it reads its input from two or three XML files and parses the files to internal tree structures. This Module can be further decomposed based smaller functionalities like file reader, parser etc.

2. Tree Matcher: The main objective of this Module is to match the tree structure. Defining a truly good matching algorithm that makes correct matches and is sufficiently fast is a research topic on its own. The matching algorithm presented here can have several desirable characteristics:

- It is reasonably fast
- Actual runs performed on the merge and use cases indicate that the matches generated are usually well-formed
- It illustrates how to implement a matcher with typed, i.e. content and structural, matching.

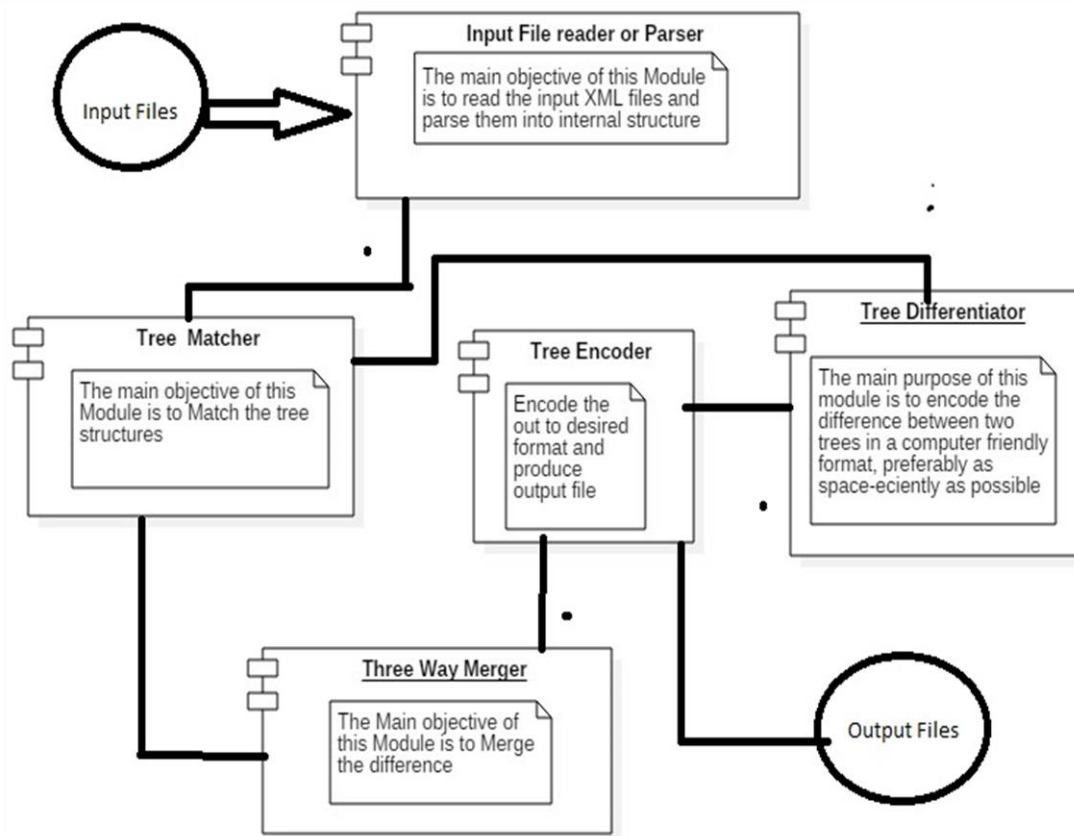


Figure 1: Architecture of 3-Way Merge Engine

3. Three-way Merger: The main objective of this Module to produce the merge tree. Unlike existing structural merging algorithms our merging algorithm directly uses the matching's MB1 and MB2, produced in the tree matching stage, to produce the merged tree. As no edit script is generated, there are no restrictions imposed by edit script operations. Neither is there a danger of strange behavior due to minimal edit script peculiarities, as described in [CG97b]. The only restriction imposed is that the matchings' are natural, the most notable implication of which is that the uncopy operation is not allowed.

4. Tree Differentiator: The main purpose of this Module is to encode the difference between two trees in a computer friendly format, preferably as space-efficiently as possible. We want the diff module to be reasonably fast and encode the differences in a manner that conserves space. We are not concerned about the diff making

sense in terms of the differences a human user would perceive between the trees, as such user-friendly differencing can be accomplished by using the 3-way merge algorithm

VI. Conclusion

The merging algorithm exhibited very few problems, things that could be improved are mainly issues regarding node context locking policies. Quite surprisingly, the heuristic matcher also performed very well, exhibiting only a few non well-formed matchings'. This observation is quite interesting considering all effort that has been put into producing optimal tree matchers. The main cause of problems turned out to be integrating edits in different branches that are close to another in a parent-child sense, giving rise to the notion of some sort of context locking along the parent-child axis in addition to the context locking inside child lists.

References

- [1]. Berliner B. "CVS II: Parallelizing software development." Proceedings of the winter 1990 USENIX Conference (Jan 22-26, 1990, Washington DC, USA), pp. 341-352. USENIX Association. Berkeley, CA, USA. Jan 1990
- [2]. Cederqvist P. et al. "Version Management with CVS." Signum Support AB, Linköping, Sweden, 1993, 162 p. Available electronically from <http://www.loria.fr/~molli/cvs/doc/cvs.pdf>. [Referred 2001/3/16]
- [3]. Groove Networks inc. "User's guide for Groove." <http://www.groove.net/pdf/usersguide.pdf> [referred 2001/3/16]
- [4]. Groove Networks inc. "Introduction to Groove." <http://www.groovenetworks.com/pdf/introducinggroove.pdf> [referred 2001/3/16]
- [5]. Lotus Development corp. "Inside Notes: The Architecture of Notes and the Domino Server." Lotus Development corp, Cambridge, MA, USA, 2000, 213 p. Available at [http://doc.notes.net/uafiles.nsf/docs/inside-notes/\\$File/insidenotes.pdf](http://doc.notes.net/uafiles.nsf/docs/inside-notes/$File/insidenotes.pdf) [referred 2001/9/10]
- [6]. Novell inc. "GroupWise 5.5 User's Guide for Windows 95/98/NT (32-bit)." <http://www.novell.com/documentation/lg/gw55/pdfdoc/55guide.pdf> [referred 2001/3/26]
- [7]. SyncML Initiative ltd. "SyncML sync protocol, version 1.0." http://www.syncml.org/docs/syncml_protocol_v10_20001207.pdf [referred 2001/3/16]
- [8]. Goland Y. et al. RFC 2518: HTTP extensions for distributed authoring [WebDAV." IETF, Reston VA, USA. Feb 1999 <http://www.ietf.org/rfc/rfc2518.txt> [referred 2001/3/19]
- [9]. Myers E. W. "An O(ND) Difference Algorithm and its Variations." *Algorithmica* vol. 1 no. 2, 1986, pp. 251-266
- [10]. Zhang K and Shasha D. Approximate tree pattern matching. In "Pattern matching in strings, trees and arrays", A. Apostolico and Z. Galil (eds.). Oxford University Press, 1997, pp. 341-371
- [11]. Berk E. "HtmlDiff: A Differencing Tool for HTML Documents." Student Project, Princeton University, Princeton, NJ, USA. 1996
- [12]. Jacobson G and Vo K-P. "Heaviest Increasing/Common Subsequence Problems." *Proceedings of the 3rd Annual Symposium of Combinatorial Pattern Matching*, Vol. 64, 1992.
- [13]. IBM Alphaworks. XML TreeDiff home page. <http://www.alphaworks.ibm.com/tech/xmltreediff> [referred 2001/9/10]
- [14]. Tai K-C. "The tree-to-tree correction problem", *Journal of the ACM*, vol. 26 no 3, 1979, pp. 422-433
- [15]. IBM Alphaworks. "XMLTreeDiff Update Languages." part of the XML TreeDiff download at <http://www.alphaworks.ibm.com/tech/xmltreediff> [referred 2001/9/10]
- [16]. Chawathe S. S, Rajaraman A, Garcia-Molina H. and Widom J. "Change detection in hierarchically structured information." In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data* (June 1996, Montreal, Quebec, Canada). pp. 493-504
- [17]. IBM Alphaworks. XML diff and merge tool home page. <http://www.alphaworks.ibm.com/tech/xmldiffmerge> [referred 2001/9/10]
- [18]. Brisson D. Statement on 1/19/1999 in the discussion forum for the XML diff and merge tool. The discussion can be accessed at <http://www.alphaworks.ibm.com/tech/xmldiffmerge> [referred 2001/9/10]