SAP DB WebDAV as Storage for (XML) Documents

Markus Özgen, Daniel Kirmse

SAP AG
Germany

Abstract

WebDAV is a recognized standard HTTP access protocol for stored documents. Since you can access these stored documents with any HTTP browser, their processing is location- and platform-independent. Also, multiple users can access the same document simultaneously.
The integration options available with web services have created a big need for storing information in openly structured XML documents and their easy retrieval. Besides texts, these documents can be of many different types like data documents and descriptions of business objects.

A database system meets the necessary technical requirements for administering a large volume of documents quickly and securely, while also letting multiple users work on them simultaneously. In the following, we present the database-based development of a WebDAV storage solution, with particular support for the storage and retrieval of XML documents.

1. What Does WebDAV Offer?

WebDAV, which stands for *Web-based Distributed Authoring and Versioning*, provides a set of enhanced HTTP protocols for administering documents on a remote Web server, and for editing these documents collaboratively. WebDAV calls directories and files *resources*, and offers you methods for creating, reading, renaming, deleting, and locking these resources. Other methods also allow you to create, change, and show *properties* for these resources. These properties can be predefined by the WebDAV server, or defined by the users. The resource locks are a simple way of enabling users to edit documents collaboratively. The property methods give you a way of assigning attributes to documents, and then using these attributes to retrieve documents. The following is a list of the WebDAV methods:

- OPTIONS Lists the supported methods
- HEAD Gets resource information
- GET Gets a resource
- PUT Stores a resource
- MKCOL Creates a collection
- PROPFIND Finds a property
- PROPPATCH Sets a property
- DELETE Deletes a resource
- COPY Copies a resource
- MOVE Renames/moves a resource
- LOCK, UNLOCK Locks/unlocks a resource

As well as these core WebDAV functions, other WebDAV enhancements exist, such as DeltaV for creating versions, or WebDAV ACL (Access Control Lists) for making detailed security settings for resources.

With these enhancements, WebDAV offers an ideal access protocol for a Web-based remote file system. Remote file systems of this type are ideal for multiple users working collaboratively in different locations.

2. Existing Application Areas

Many existing applications can use WebDAV-based storage to exchange data. These include applications such as Microsoft Explorer, which enables you to use a WebDAV Server as a Web folder and access it as a regular disk drive. Microsoft Office 2000 lets you use these Web folders directly in the file-open and save dialogs. Microsoft Office 2000 also uses the WebDAV locks to enable multiple users to edit documents simultaneously.

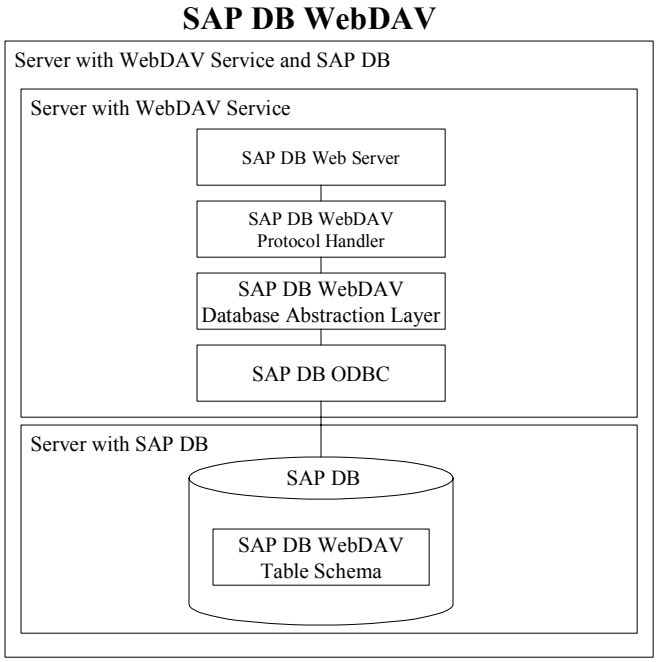
On Unix, Java-based WebDAV clients also exist that allow you to access documents stored on a WebDAV Server.

Among other things, SAP uses SAP DB WebDAV for sharing documents remotely, or for archiving documents, including XML documents.

3. Why Use a Database to Store Documents?

A database already includes much of the required technology for developing a WebDAV-based document storage, and does not store documents and their properties independently. Unlike a file system, a database includes the creation or deleting of a resource as well as its properties in a single transaction. If an error occurs, the entire transaction is rolled back. Documents cannot exist without properties, just as properties cannot exist without an accompanying document. Employing simple SQL methods, you can use the documents or properties with other relational data. Nowadays, database backups are relatively simple and quick, since the backup function is a fundamental database component. Comprehensive database backup tools exist that are also well-supported by hardware. In the same way, it is much simpler and quicker to recover a database using its restore function than it is to restore a file system. Databases are well-suited to administering large volumes of data, such as directories with several million documents, which would push a file system to its limits.

4. The Architecture of SAP DB WebDAV



4.1 Table Schema

To implement a document storage in a database, you must first map a type of file system using relational data. Documents are usually located in a path (`/MyDocuments/Private/WebDAV.doc`, for example). Each node in a path has a parent node. The first node in the path is the root node, to which all other nodes are connected. The documents themselves are located either directly under the root node, or under one of the other nodes in the path. These relationships are known as parent-child relationships, and are stored in a node table in the database.

Alongside the file system, the properties of the path nodes and documents must also be administered. These properties are name-value pairs and are stored in a property table. To improve performance when you access these property values, they are split into short and long values. If the value is short (up to a certain key length), it is kept in a key column. If it is longer, it is located in a BLOB (Binary Large Object).

The documents themselves, i.e. their content, are stored as BLOBs in a separate content table, and are also split into different columns depending on whether the content is short or long, for performance reasons. You also have the option of compressing the document content before it is stored.

4.2 ODBC (Open Database Connectivity) as Database Interface

As well as an appropriate table schema, you also need an interface for the database. Since SAP DB WebDAV is implemented in C, the standard SAP DB call interface, ODBC, is the most suitable solution. The SAP DB ODBC driver is optimized for use with SAP DB and supports ODBC Core Levels 2 and 3. The SAP DB ODBC driver also supports the full ODBC 3.5 API, as well as the Microsoft ODBC add-ons for supporting 64 bit platforms.

4.3 Database Abstraction Layer

A database abstraction layer on top of the ODBC interface bundles the interface and schema away from the next higher layer, i.e. the WebDAV protocol handler.

4.4 WebDAV Protocol Handler

The WebDAV protocol handler maps the current WebDAV protocol to the database abstraction layer. When the protocol handler receives a WebDAV request, it

identifies the WebDAV method, accordingly requests a database connection, and uses this connection to communicate with the database abstraction layer. The WebDAV protocol handler is then also responsible for composing and sending the reply to the WebDAV request.

4.5 The SAP DB Web Server

Since WebDAV is an HTTP enhancement, the WebDAV requests are sent to the WebDAV protocol handler through a Web server. In our case, this can be either our own SAP DB Web Server, or an Apache Web server. First, the Web server receives an inbound HTTP request, and analyzes the URL to see whether it is a WebDAV request. If so, the request is forwarded to the WebDAV protocol handler.

The SAP DB Web Server offers database connection handling functions that are optimized for SAP DB. These include, among other things, a function for managing database connection pools (session pools) that, after processing a WebDAV request, gives the WebDAV protocol handler the option of returning open database sessions to the pool, where they are then immediately available for the next WebDAV request. Along with the database session, other reusable program resources, particularly prepared statements, are also pooled.

5. How Are XML Documents Stored?

Essentially, XML documents are just documents that are stored as BLOBs. However, since their content is structured, and the structure is maybe even described by a structure definition (such as DTD (Document Type Definition) or an XML schema), they can be analyzed easily by programs. This makes it simple to extract information from these structured XML documents and make it available to other applications. Apart from the BLOBs, which contain the entire XML document, this extracted information is kept in regular tables. Foreign key relationships link this information to the documents. In this way, for example, delete operations in documents also delete the extracted information, which keeps the data consistent. SAP DB WebDAV uses extracted information to index XML documents.

6. Why Index XML Documents?

It is a general requirement that documents can be retrieved easily from a large number of stored documents. Unstructured documents can be indexed with a full text index, and the index values used to retrieve the documents. These index values are very imprecise when you search for documents. For example, if the name *Marc* is indexed in an unstructured document, then you have no context for the name in this document. If you now search for documents *authored* by *Marc*, then you also find documents that use the name *Marc* as an example, or include information about *Marc*. XML documents, on the other hand, use their structured form to specify the author precisely. For example, the value *Marc* is in the XML path `/DBTechnology/WebDAV/Author`. If you store the XML path `/DBTechnology/WebDAV/Author` as a property of this document with the value *Marc*, you will later be able to find only those documents that have this author as a value in this XML path.

Another reason for indexing XML documents is to make their content available to other applications. For example, if XML documents describe business objects, such as a purchase order, then the information in the documents can be used to trigger an order request. The Web service that received the purchase order would send it as an XML document (with a standardized XML structure for purchase orders) to another Web service, where the order is processed. These two Web services could use different technologies, but exchange the purchase order in that structured XML document. Once the purchase order reaches the order processor, and is added to the stored orders, all the information required to process the order could be extracted from the XML document by the indexing and made available to a back-end system.

7. SAP DB WebDAV XML Indexing

SAP DB WebDAV can store any type of document or file, and use the WebDAV protocol to administer them. Alongside the administration of documents, SAP DB WebDAV also gives you the option of using document classes to index XML documents. A browser-based application is used for the administration of SAP DB WebDAV and for the administration of the document classes, the index definitions, and the extension mappings. This application also allows you to search for indexed XML documents.

SAP DB WebDAV can index documents either synchronously or asynchronously. Asynchronous indexing uses a separate indexing engine that can be distributed across multiple servers to improve scalability. SAP DB WebDAV makes use of the document classes for its indexing. These classes include index definitions in which one or more XPath location paths describe the XML paths that need indexing. Each XPath location path is assigned a unique index name, which can then be reused in different document classes. When the documents are indexed, the index values are entered with this index name as name-value pairs in the property table. These pairs are then available as WebDAV properties of the XML document. A SAX (Simple API for XML) parser is used for the indexing.

7.1 Document Classes

Essentially, SAP DB WebDAV stores XML documents in the same way as regular documents. However, XML documents can be assigned a property that defines their document class. This class then defines how the XML documents are indexed. There are several different ways of communicating to the SAP DB WebDAV Server which document class to assign to which XML document: the property of the XML document can be set directly with WebDAV methods, the document class can be specified with special HTTP headers, or you can use the extension mapping. In the latter case, it is specified in advance which document name extension corresponds to which document class. If you use the special HTTP header, you can also decide whether the document is indexed synchronously when it is stored, or asynchronously.

7.2 Synchronous Indexing

If you want to index the XML document synchronously, then an appropriate HTTP header must be sent when the document is stored. The document class must either accompany this HTTP header, or be determined from an entry in the extension mapping table. The XML document is stored and indexed in the same transaction, which means that if it cannot be indexed, then it cannot be stored. In the same way, if it cannot be stored, then it is not indexed. Once it has been indexed successfully, the XML document is flagged as such.

7.3 Asynchronous Indexing

If you want to index an XML document asynchronously, then the XML document is not flagged as indexed when it is stored or updated. If the asynchronous indexing process finds an XML document that is not yet flagged as indexed, and a document class exists for this XML document, then the process indexes the XML document. The document class either accompanies a special HTTP header when the document is stored, or you use the WebDAV property for the document class retrospectively. If a fixed document class is specified, then this is used for the indexing; if not, then an attempt is made to determine the document class from the document extension and extension mapping table.

7.4 Re-Indexing

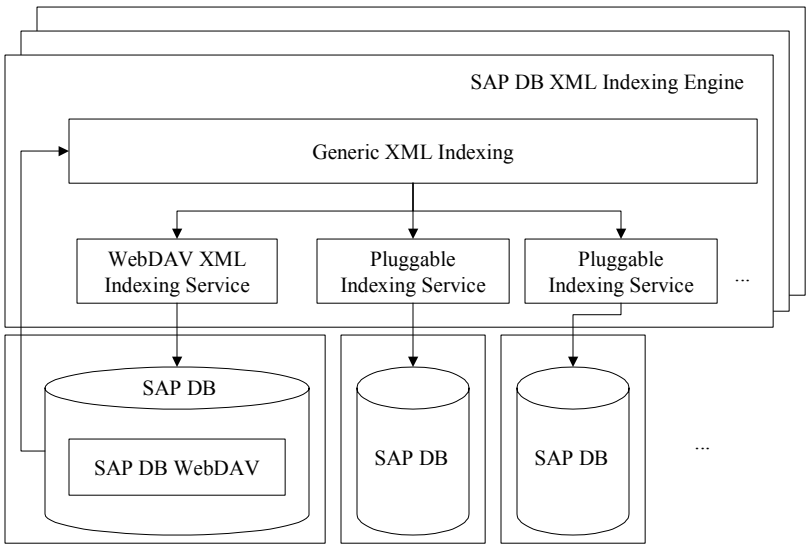
If the document classes, index definitions, extension mapping table, or the WebDAV property for the document class have been modified, then the documents are re-indexed, asynchronously. The documents can be re-indexed at any time. If the document classes, index definitions, or the extension mapping table have been modified, then the index values of the affected XML documents are deleted, and the documents are flagged as not being indexed. If the WebDAV property for the document class of an XML document has been modified, then only the index values of this XML document are deleted, and the document flagged as not being indexed.

7.5 Retrieving XML Documents

The index values that are extracted when XML documents are indexed are stored as WebDAV properties of the particular XML document. This means that the WebDAV method PROPFIND can use index names, which correspond to specific XPath location paths, to retrieve XML documents. Since the index values are stored as relational data in the property table, the relevant XML documents can also be retrieved directly with SQL methods. SAP DB WebDAV provides the browser-based XML Document Query Service to enable this. You can use this service to formulate search queries that specify a range of document classes and index names, and the corresponding index values. You can also use Boolean expressions to link these search conditions.

8. The Indexing Engine

SAP DB WebDAV XML Indexing



The asynchronous indexing engine is a multithreaded application. Multiple indexing engines can index documents simultaneously in the same SAP DB WebDAV. Since this means that more than one thread or indexing engine searches for documents in SAP DB WebDAV in parallel, they cannot be allowed to block each other. Special non-blocking SQL selects make sure that this cannot happen. If a select comes across a data record that is blocked because it is being processed by another indexing thread, then the select does not persist until the block is removed; instead, it skips this data record, and searches for other records that are relevant for indexing. If it finds another data record, then this data record is blocked, and subsequent selects cannot process it, but neither do they persist with it. If the data record is released due to a termination (such as a rollback caused by a program termination), then it is selected again later by a subsequent indexing thread, which makes a new attempt at indexing it. This means that XML documents are not indexed in any particular order.

The SAP DB WebDAV indexing engine is constructed with a generic kernel that is responsible for searching for the XML documents and distributing the resulting indexing requests. This means that different indexers can be called dynamically to process the indexing requests. The indexer that indexes the XML documents in the SAP DB WebDAV property table is already provided with SAP DB WebDAV. If you want to store the index entries in a schema other than the property table, then you can use a special self-defined indexer to do this. The information about which indexer is responsible for which document class is also administered in the browser-based application mentioned above.

9. SAP DB WebDAV Availability

As of Version 7.4.03, SAP DB WebDAV is available for the following platforms (operating system/processor type):

- Compaq Tru64 Unix /Alpha
- IBM AIX /PowerPC
- SUN Solaris /SPARC
- HP-UX /HP-PA
- Linux /Intel
- Windows NT /Intel
- Windows 2000 /Intel

SAP DB WebDAV, including the XML indexing function, is available as open source software at www.sapdb.org where you can also find additional information, documentation and other SAP DB downloads.