Web-based Distributed Authoring and Versioning (WebDAV)

Mike Gartrell
November 16, 2006
Agenda

- WebDAV goals
- Functionality
- Concepts
- Operations
- WebDAV operation examples
- Demo
- Is WebDAV RESTful?
The stated goal of the WebDAV IETF working group, from the group’s charter: “define the HTTP extensions necessary to enable distributed web authoring tools to be broadly interoperable, while supporting user needs”.

The WebDAV protocol’s aim was to make the Web a writable, collaborative medium, in line with Tim Berners-Lee’s original vision.
Goals (2 of 2)

- From the DAV FAQ: “… people working on DAV have had goals which extend beyond simple web page authoring.”
- Additional goals (from the FAQ):
  - A network filesystem for the Internet, that operates on entire files at a time, and performs well in high-latency environments.
  - A protocol for manipulating the contents of a document management system via the Web.
  - Support of remote software development teams.
  - Leverage the success of HTTP in being a standard access layer for a wide range of storage repositories. HTTP provides read access, while DAV provides write access.
Functionality

- Locking: Both exclusive and shared write locks are supported. Locks address the overwrite problem, where two or more editors write to the same resource without first merging changes.
- Properties: Metadata, persisted as XML, are associated with each resource.
- Namespace manipulation: Resources may be copied or moved. Collections (similar to directories) may be created and listed.
- Versioning support was left out of WebDAV, and was later defined by the DeltaV specification.
WebDAV Concepts

- Resources
- Properties
- Collections
- Locks
WebDAV Concepts: Resources

- The concept of a resource is the same as that found in HTTP. A resource is the referent ("target") of any URI.
WebDAV Concepts: Properties

- A property is a name-value pair that contains descriptive information (metadata) about a resource.
- Properties are persisted as XML.
WebDAV Concepts: Collections

- A collection is a resource that contains a set of URIs, called member URIs, which identify member resources.
- A collection is essentially a directory within the server’s namespace.
WebDAV Concepts: Locks

- Locks provide a mechanism for serializing access to a resource. WebDAV defines locking only for write access to a resource.
- WebDAV supports two kinds of locks on a resource: exclusive and shared.
  - Exclusive: lock is only granted to a single user. Eliminates the need to merge changes from multiple users.
  - Shared: multiple users may receive the lock. The shared lock informs other collaborators that a user may be working on a resource. Implies that this user trusts that his collaborators will not overwrite his work.
WebDAV Operations

- WebDAV adds the following methods to HTTP:
  - PROPFIND – retrieve properties from a resource. Also overloaded to allow retrieval of the contents of a collection.
  - PROPPATCH – set and/or remove properties on a resource.
  - MKCOL – create a collection (directory).
  - COPY – copy a resource from one URI to another.
  - MOVE – move a resource from one URI to another.
  - LOCK – put a lock on a resource.
  - UNLOCK – remove a lock from a resource.
Retrieve the specified properties for the 
http://www.foo.bar/file resource:

- Sample request:

```xml
PROPFIND /file HTTP/1.1
Host: www.foo.bar
Content-type: text/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8" ?>
<D:propfind xmlns:D="DAV:"
  xmlns:R="http://www.foo.bar/boxschema/">
  <D:prop>
    <R:bigbox/>
    <R:author/>
    <R:DingALing/>
    <R:Random/>
  </D:prop>
</D:propfind>
```
Sample response:
HTTP/1.1 207 Multi-Status
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8"?>
<D:multistatus xmlns:D="DAV:">
  <D:response>
    <D:propstat>
      <D:prop xmlns:R="http://www.foo.bar/boxschema/">
        <R:bigbox><R:BoxType>Box type A</R:BoxType></R:bigbox>
        <R:author><R:Name>J. J. Johnson</R:Name></R:author>
      </D:prop>
      <D:status>HTTP/1.1 200 OK</D:status>
    </D:propstat>
    <D:propstat>
      <D:prop><R:DingALing/><R:Random/></D:prop>
      <D:status>HTTP/1.1 403 Forbidden</D:status>
      <D:responsedescription> The user does not have access to the DingALing property. </D:responsedescription>
    </D:propstat>
  </D:response>
  <D:responsedescription> There has been an access violation error. </D:responsedescription>
</D:multistatus>
Set the specified properties for the http://www.foo.com/bar.html resource:

- Sample request:
  
  PROPPATCH /bar.html HTTP/1.1  
  Host: www.foo.com  
  Content-Type: text/xml; charset="utf-8"  
  Content-Length: xxxx

  <?xml version="1.0" encoding="utf-8" ?>  
  <D:propertyupdate xmlns:D="DAV:" xmlns:Z="http://www.w3.com/standards/z39.50/">  
  <D:set>  
    <D:prop>  
      <Z:authors>  
        <Z:Author>Jim Whitehead</Z:Author>  
        <Z:Author>Roy Fielding</Z:Author>  
      </Z:authors>  
    </D:prop>  
  </D:set>  
  <D:remove>  
    <D:prop><Z:Copyright-Owner/></D:prop>  
  </D:remove>  
  </D:propertyupdate>
WebDAV Operation Examples: PROPPATCH (2 of 2)

- Sample response:

HTTP/1.1 207 Multi-Status
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx
<?xml version="1.0" encoding="utf-8" ?>
<D:multistatus xmlns:D="DAV:" xmlns:Z="http://www.w3.com/standards/z39.50">
  <D:response>
    <D:propstat>
      <D:prop><Z:Authors/></D:prop>
      <D:status>HTTP/1.1 424 Failed Dependency</D:status>
    </D:propstat>
    <D:propstat>
      <D:prop><Z:Copyright-Owner/></D:prop>
      <D:status>HTTP/1.1 409 Conflict</D:status>
    </D:propstat>
    <D:responsedescription>Copyright Owner can not be deleted or altered.</D:responsedescription>
  </D:response>
</D:multistatus>
WebDAV Operation Examples: MKCOL

- Create the
  http://www.server.org/webdisc/xfiles/ collection:

  - Sample request:
    MKCOL /webdisc/xfiles/ HTTP/1.1
    Host: www.server.org

  - Sample response:
    HTTP/1.1 201 Created
WebDAV Operation Examples: COPY

- Copy the resource http://www.ics.uci.edu/~fielding/index.html to the location http://www.ics.uci.edu/users/f/fielding/index.html, overwriting the existing resource at the destination:
  - Sample request:
    COPY /~fielding/index.html HTTP/1.1
    Host: www.ics.uci.edu
  - Sample response:
    HTTP/1.1 204 No Content
WebDAV Operation Examples: MOVE

- Move the resource
  http://www.ics.uci.edu/~fielding/index.html to the location
  http://www.ics.uci.edu/users/f/fielding/index.html (there is initially nothing at the destination resource):
  - Sample request:
    MOVE /~fielding/index.html HTTP/1.1
    Host: www.ics.uci.edu
  - Sample response:
    HTTP/1.1 201 Created
WebDAV Operation Examples: LOCK (1 of 2)

- Create an exclusive write lock on the resource
  http://webdav.sb.aol.com/workspace/webdav/proposal.doc

  - Sample request:
    LOCK /workspace/webdav/proposal.doc HTTP/1.1
    Host: webdav.sb.aol.com Timeout: Infinite, Second-4100000000
    Content-Type: text/xml; charset="utf-8"
    Content-Length: xxxx
    Authorization: Digest username="ejw", realm="ejw@webdav.sb.aol.com",
      nonce="...", uri="/workspace/webdav/proposal.doc", response="...",
      opaque="..."

    <?xml version="1.0" encoding="utf-8" ?>
    <D:lockinfo xmlns:D='DAV:'>
      <D:lockscope><D:exclusive/></D:lockscope>
      <D:locktype><D:write/></D:locktype>
      <D:owner>
      </D:owner>
    </D:lockinfo>
WebDAV Operation Examples: LOCK (2 of 2)

- Sample response:

HTTP/1.1 200 OK
Content-Type: text/xml; charset="utf-8"
Content-Length: xxxx

<?xml version="1.0" encoding="utf-8" ?>
<D:prop xmlns:D="DAV:">
  <D:lockdiscovery>
    <D:activelock>
      <D:locktype><D:write/></D:locktype>
      <D:lockscope><D:exclusive/></D:lockscope>
      <D:depth>Infinity</D:depth>
      <D:owner>
        <D:href>
          http://www.ics.uci.edu/~ejw/contact.html
        </D:href>
      </D:owner>
      <D:timeout>Second-604800</D:timeout>
      <D:locktoken>
        <D:href>
         opaqueLocktoken:e71d4fae-5dec-22d6-f0a0c91e6be4
        </D:href>
      </D:locktoken>
    </D:activelock>
  </D:lockdiscovery>
</D:prop>
WebDAV Operation Examples: UNLOCK

- Remove the lock, identified by the specified lock token, from the resource
  http://webdav.sb.aol.com/workspace/webdav/info.doc

  - Sample request:
    UNLOCK /workspace/webdav/info.doc HTTP/1.1
    Host: webdav.sb.aol.com
    Lock-Token: <opaquelocktoken:a515cfa4-5da4-22e1-f5b5-00a0451e6bf7>
    Authorization: Digest username="ejw", realm="ejw@webdav.sb.aol.com",
    nonce="...", uri="/workspace/webdav/proposal.doc", response="...",
    opaque="...

  - Sample response:
    HTTP/1.1 204 No Content
Sources

- RFC 2518: HTTP Extensions for Distributed Authoring
  - WebDAV (http://webdav.org/specs/rfc2518.html)
- WebDAV article on Wikipedia (http://en.wikipedia.org/wiki/WebDAV)
- DAV FAQ (http://webdav.org/other/faq.html)
- Some WebDAV extensions (not discussed in this presentation):
  - RFC 3253: Versioning Extensions to WebDAV (DeltaV; http://www.webdav.org/specs/rfc3253.html)
Demo

- WebDAV server: Jakarta Slide (http://jakarta.apache.org/slide)
- WebDAV client: DAV Explorer (http://www.davexplorer.org)
Is WebDAV RESTful?

- Yes. WebDAV is an HTTP extension, and it simply defines new verbs for HTTP.
- However, Roy Fielding has commented that “the PROP* methods conflict with REST because they prevent important resources from having URIs and effectively double the number of methods for no good reason.” See an archive of his post at http://www.mail-archive.com/microformats-rest@microformats.org/msg00189.html
- Do any of WebDAV’s new methods violate the principles of REST? Thoughts?