

Filtering Content for Customized User Assistance

Sarah O'Keefe
DocTrain 2007
Lowell, MA



scripkeefe.com

Agenda

- Content reuse scenarios
- Traditional versioning techniques
- Bringing XML into the picture



scripkeefe.com

Why this topic for me?

- Publishing consultant.
- Versioning always a concern for clients.
 - Frustrated with conditional text limitations.
 - Interested in finding more powerful alternatives.
- XML provides an elegant solution, but requires significant implementation planning and effort.



scripkeefe.com

Content reuse scenarios

- Baseline/superset
- One-off customization
- Topic assembly
- Customization per user or role



scripkeefe.com

Baseline/superset

- Baseline version for common text
- Modified version 1 contains baseline+
- Modified version 2 contains baseline+
- Minimal overlap between version 1 and version 2 information



scripkeefe.com

One-off customization

- Baseline is used for normal deliverables
- One client requests customization
- Writers make copy of baseline, deliver changes
- Custom version is not maintained



scripkeefe.com

Topic assembly

- Content library contains modules
- Each deliverable uses some of the modules
- Modules are the same for each deliverable
- Topic assembly is manual
 - Identify relevant topics
 - Create topic lists for each deliverable



Topic assembly

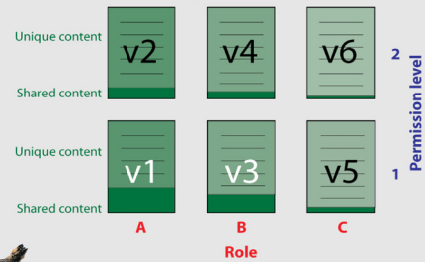


Customization per user or role

- Baseline information is limited
- Almost all information is customized
- Content is often reused in multiple versions
- Large number of versions



Customization per user or role



Traditional versioning choices

- Conditional text
- Build tags
- Reuse



Conditional text and build tags

- Excellent for baseline/superset with no overlap among superset versions
- Problems with content that requires multiple conditions



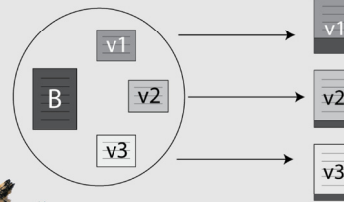
Reuse

- Suitable for topic assembly
- Need a fixed set of deliverables because topic lists must be defined ahead of time



Traditional versioning

- Customization occurs during authoring process

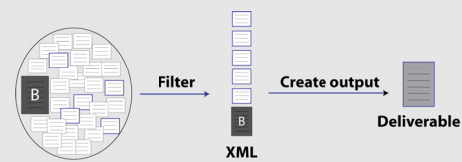


Creating user-specific content

- Number of variations is too large to manage with traditional techniques
- Examples
 - Users see different software interfaces based on their permissions
 - Mechanic assembles task list for a particular maintenance process
- Customization shifts from authoring to delivery



Customization upon user request



XML for content reuse

- Developing the content model
- Writing content for reuse
- Creating user-specific processing



Developing the content model

- Analyze requirements
- Develop elements and especially attributes to support versioning
- Most often, one attribute per content "dimension."



Analysis example 1

- User guide is customized for each client.
- Client information may overlap.
- Result:
 - Client attribute
 - Can have multiple values



Analysis example 2

- Web-based application
- Interface changes based on user role and permissions
- Result:
 - Create role and permission attributes
 - Use values that match software roles and permissions



Analysis example 3

- Training modules need to be mixed and matched to create custom courses
- Result:
 - Attributes for user level, prerequisites, duration, subject matter, instructor requirements, equipment requirements, delivery options (and probably more)
 - Sequencing is a challenge



Writing content for reuse

- Content must be modular
- Writers learn to identify reuse candidates
- Modules assembled to create deliverables

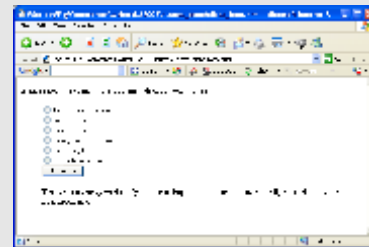


Creating user-specific processing

- Write filters based on user profile information
- Filter based on attribute values?
- Real-time versus static versioning
 - Static: easier to set up
 - Real-time: more flexible



Demo



Versioning by client

- Specify client you want
- XSL processing identifies relevant XML content
- Assembly, publishing, and formatting is done on the fly



Another demo



Filtering by client and platform

- XSL processing is more complex
- Assembly, publishing, and formatting is the same as client-only example



Summary

- XML-based versioning supports traditional, limited versioning
- XML lets you go beyond traditional options
- Excellent solution for user profiling or documentation that's customized on the fly



Summary (continued)

- Expensive to implement
- Attribute-based versioning will require custom(-ized) structure
- All XML project caveats apply



Your questions



XML resources

- Scriptorium white papers
 - www.scriptorium.com/papers.html
- XML Architecture for Customizing Content
 - www.writersua.com/articles/xml/index.html



Contact information

Scriptorium Publishing
www.scriptorium.com
info@scriptorium.com
+919.481.2701 x105

