GT2003HackathonJsr170

David Nuscheler on JSR 170

JSR 170 - Content Repository for JavaTM technology API

David is from Day software

Overview

Problem: over 800 CMS vendors, currently each with their own Solution: API. JSR 170 aims to standardize access to content repositories.

JSR 170 is an API, compared to WebDAV which is a protocol.

It sits between something like WebDAV and an actual repository.

Even though big players who could have conflicting interests were involved in the design of JSR 170, it seems like the API is clean, well designed and everybody agrees on it.

JSR 170 is entering the community review phase, an Open-Source reference implementation is in the works.

Apache members have access to the JCP "papers" but not committers.

Contains simple query language.

Workflow and lifecycle management (auto archiving etc). are seen as an applications, not as being part of the repository.

Roadmap

- Community review is imminent, should be done by the end of 2003
- Reference implementation (from Day IIUC?) with Open-Source-friendly and patent-(un)friendly license (designed to avoid people patenting parts of the implementation).

Levels of compliance

Similar in concept to the JDBC compliance levels.

Level 1: everything you need for Presentation and Publishing

Basic compliance level for storage (vendors of repositories).

- Read and write access to the content
- Hierarchy operations - hierarchical storage space like a filesystem, familiar
- Serialization/deserialization - mapping repo structure to ??
- Node Type - how does this node work, what is it's meta data?
  - no difference between data and meta-data, they are handled in the same way, not seen as "different animals"
- Search/Filter
  - a SQL-like query language, comes with the API ** based on SQL syntax, with extensions ** geared more towards hierarchies
- Linking

According to David, Level 1 contains everything you need for presentation and publishing.

Level 2: versioning, transactions, locking (a superset of Level 1)

- Versioning (David warns that you might need a brain transplant first to understand this part of the spec)
  - works with linking
  - Still needs TLC
- Transactions (distributed, XA)
- Observation (Events, similar to DB Triggers etc.) (optional part of the Interface)
  - Synchronous/Asynchronous events. Synchronous events are vetoable.
  - eg. will be very powerful for Cocoon, repo can tell Cocoon Cache about changes
  - eg. will make currently uncachable sources, cachable
- Locking

Questions/remarks

- Gianugo asks about connecting a level 1 client to a level 2 repository: will versions be created automatically?
  - Hard to answer, it will be up to the repository to implement implicit versioning or not.
  - Problem could be, server creates a new version for each save operation

Reference Implementation (proposal inside Slide)

Most of Level 1 (without searching, they will use Lucene for this)

How to move forward?

Slide never made it off the ground. (Stefano : It was 2 seperate one-man-shows) XIndice does not evolve.

Create a new community for this (in Cocoon?) where people care about this?

Question: What about Lenya? Stefano: IMO Lenya is working slightly the wrong way, mixing back and front end (, ie. mixing User and Editor view)

We should stop at the API.