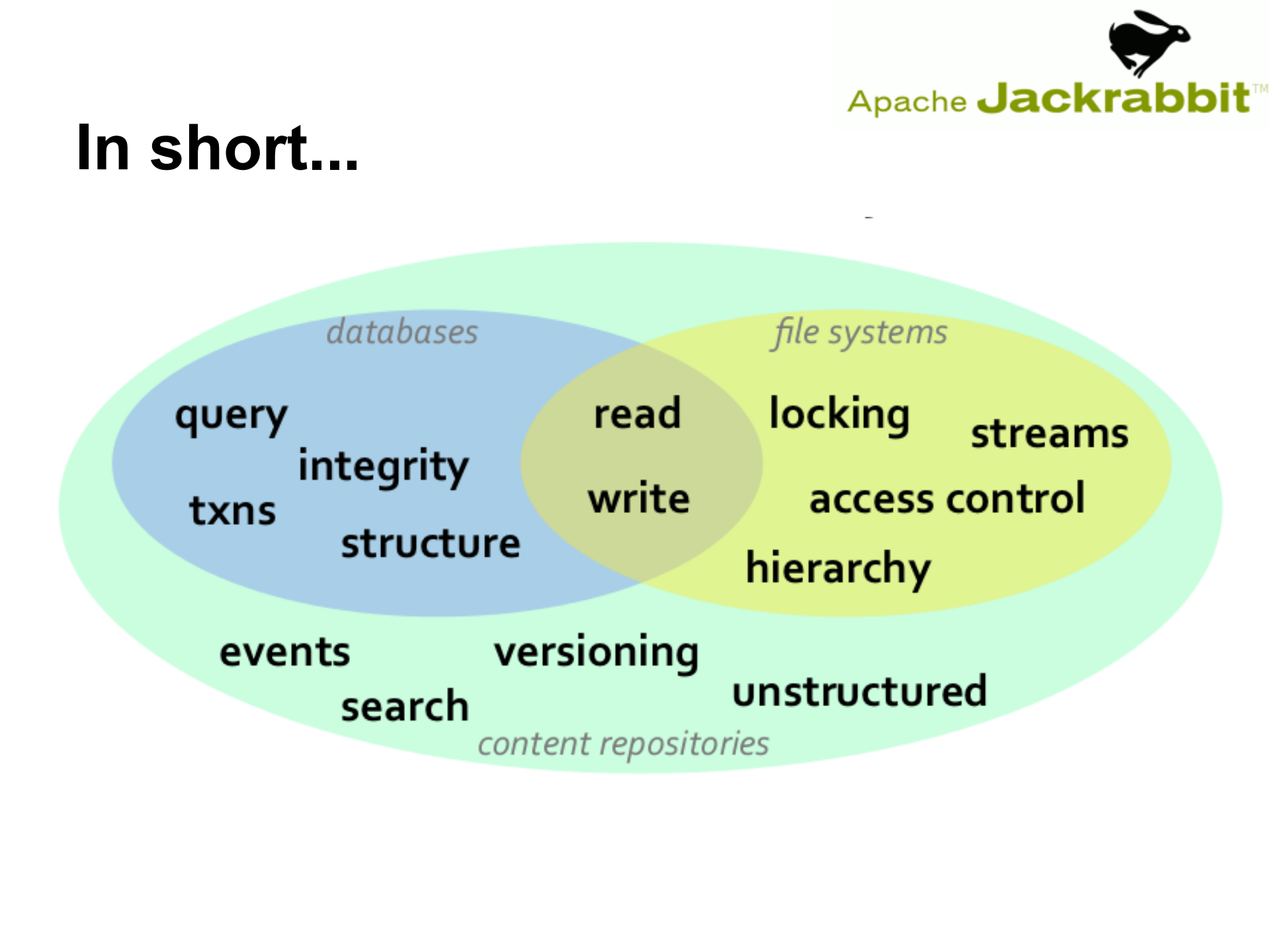
Brian Snijders

8 november 2012

In short...

- Hierarchical content store
- Versioning, transactions, ...
- Both structured and unstructured content

- Java Content Repository API:
  - JCR 1.0 (JSR-170)
  - JCR 2.0 (JSR-283)


Apache Jackrabbit™
In short...
databases
file systems
query
integrity
txns
structure
read
write
locking
streams
access control
hierarchy
events
versioning
search
unstructured
content repositories

Today's focus

- JackRabbit characteristics (NoSQL vs traditional)
- Internal architecture and operation

- Hands-on providing:
  - Basic operational knowledge on JCR repositories
  - Storage and construction of a simple, but flexible and dynamic model
  - Retrieval from this model by queries

JackRabbit as a NoSQL-store (1)

- Document-store, able to store the concept of a 'document', encapsulated in some standard format/encoding (XML)

- Documents of the same type can differ, so no uniform 'record-like' structure

JackRabbit as a NoSQL-store (2)

- Multiple lookup mechanisms
  - <K,V>-based, so Map<uuid, document>
  - XPath
  - SQL
    - :-(when strictly NoSQL
    - :-) when NotOnlySQL
  - ... and more in JCR-2.0

- Distributed data and redundancy using propagation of 'changesets'

Apache Jackrabbit™

JackRabbit as a traditional store

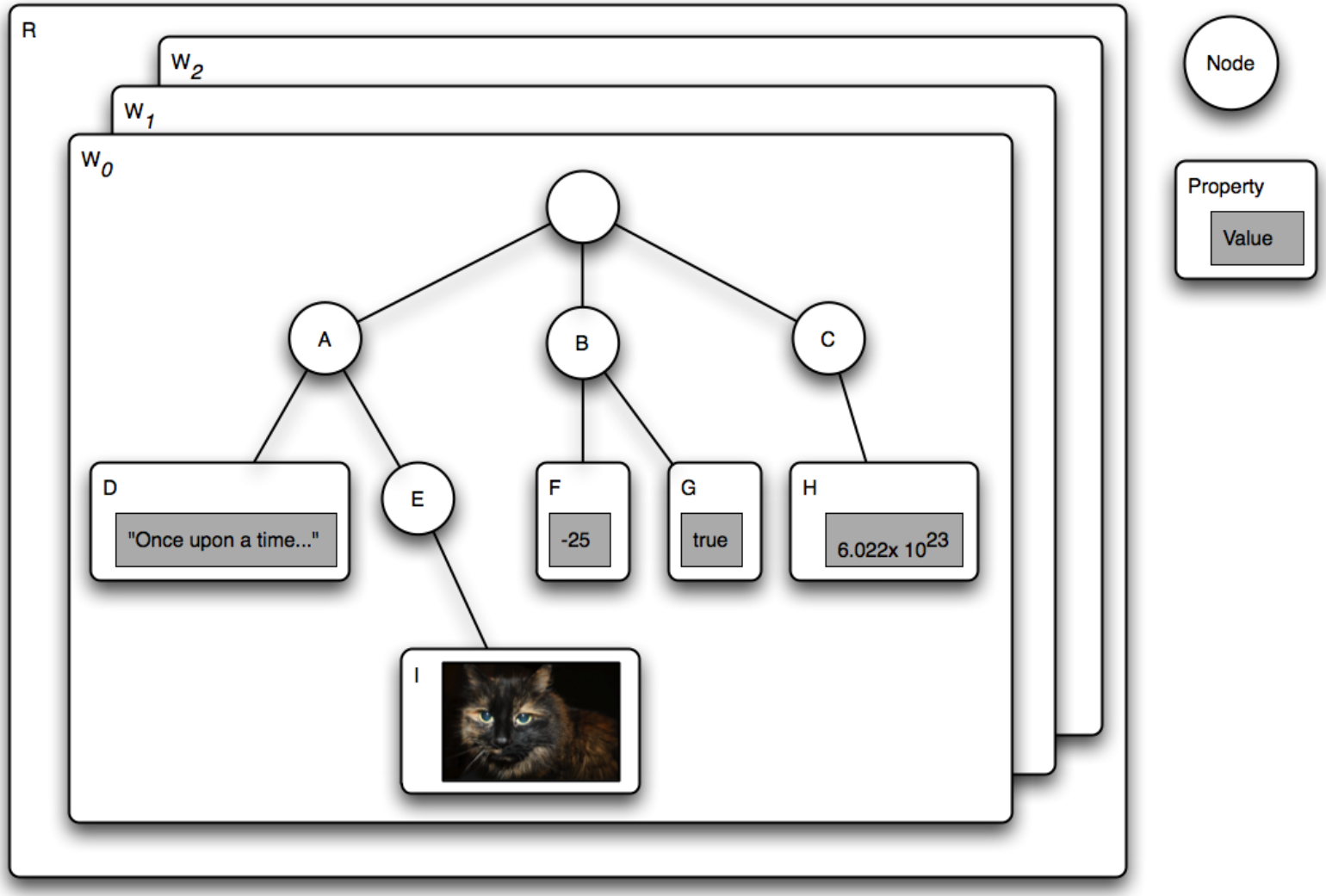
- Namespaced repository structure, yielding strict node type definitions:
  - Suggests "Restricting programmer freedom"
  - Resembles the concept of a DBMS schema

- Still backed by a relational database for persisting state management, e.g. clustering journals

- Still using SQL for convenient retrieval

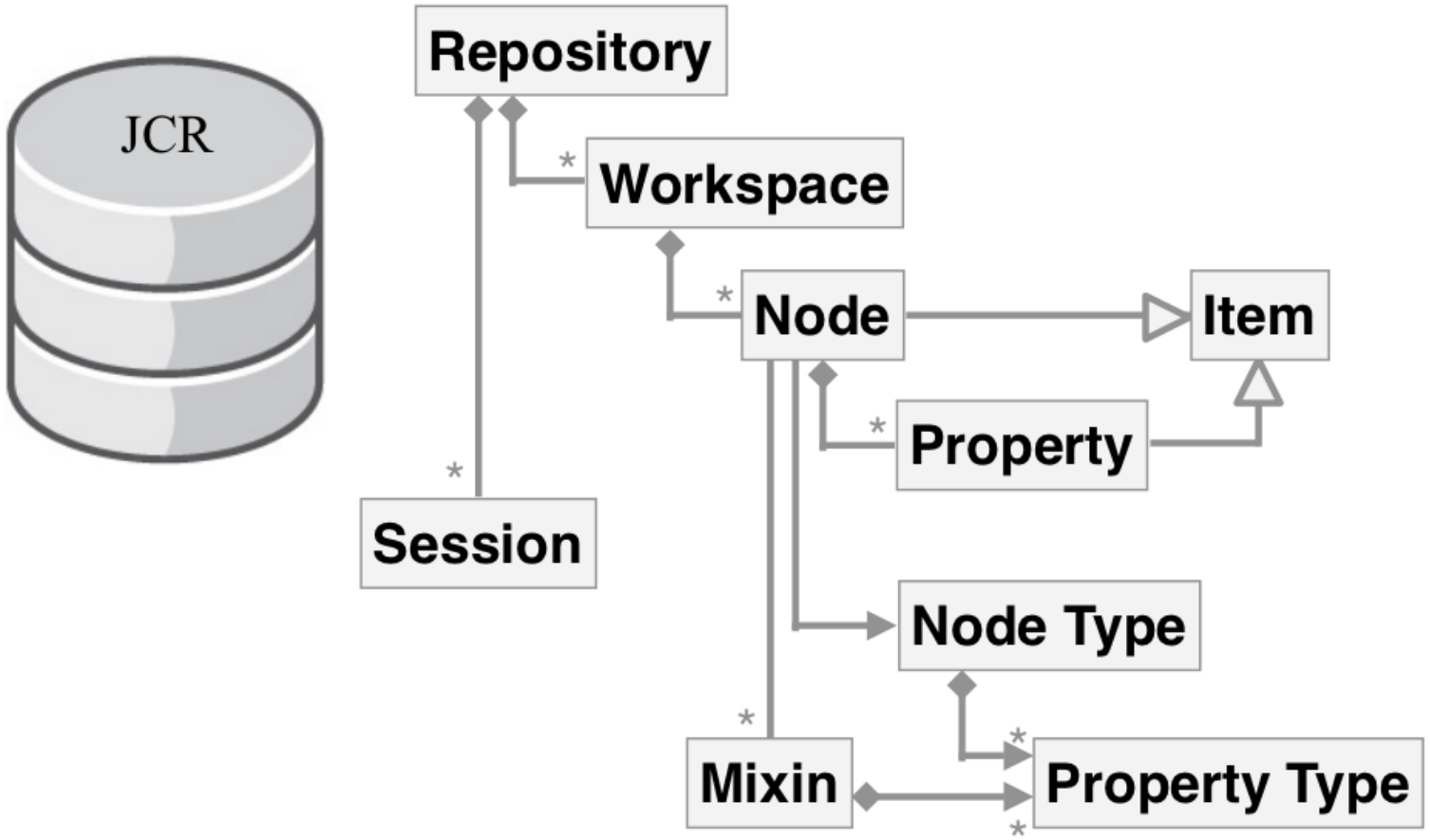
Apache Jackrabbit™

Internals - Repository



Apache Jackrabbit™

Internals - Repository



Internals - Node

- Node:
  - Inherits from or has NodeType 'nt:base'
  - Mandatory 'jcr:primaryType'
  - Non-mandatory 'jcr:mixinTypes'
- Each node
  - Node (0...n)
  - Mixins (0...n)
  - Properties (0...n)
- Identified by absolute/relative paths in the repository hierarchy
- Compact Nodetype Definition (CND)
  - Used for structured, namespaced nodes
  - Structural scoping and consistency

Apache Jackrabbit™

Internals - Mixins

"Mixin node types are used to add additional properties or child nodes to a given node instance, typically in order to expose some aspect of a specialized repository feature" (JCR 2.0)

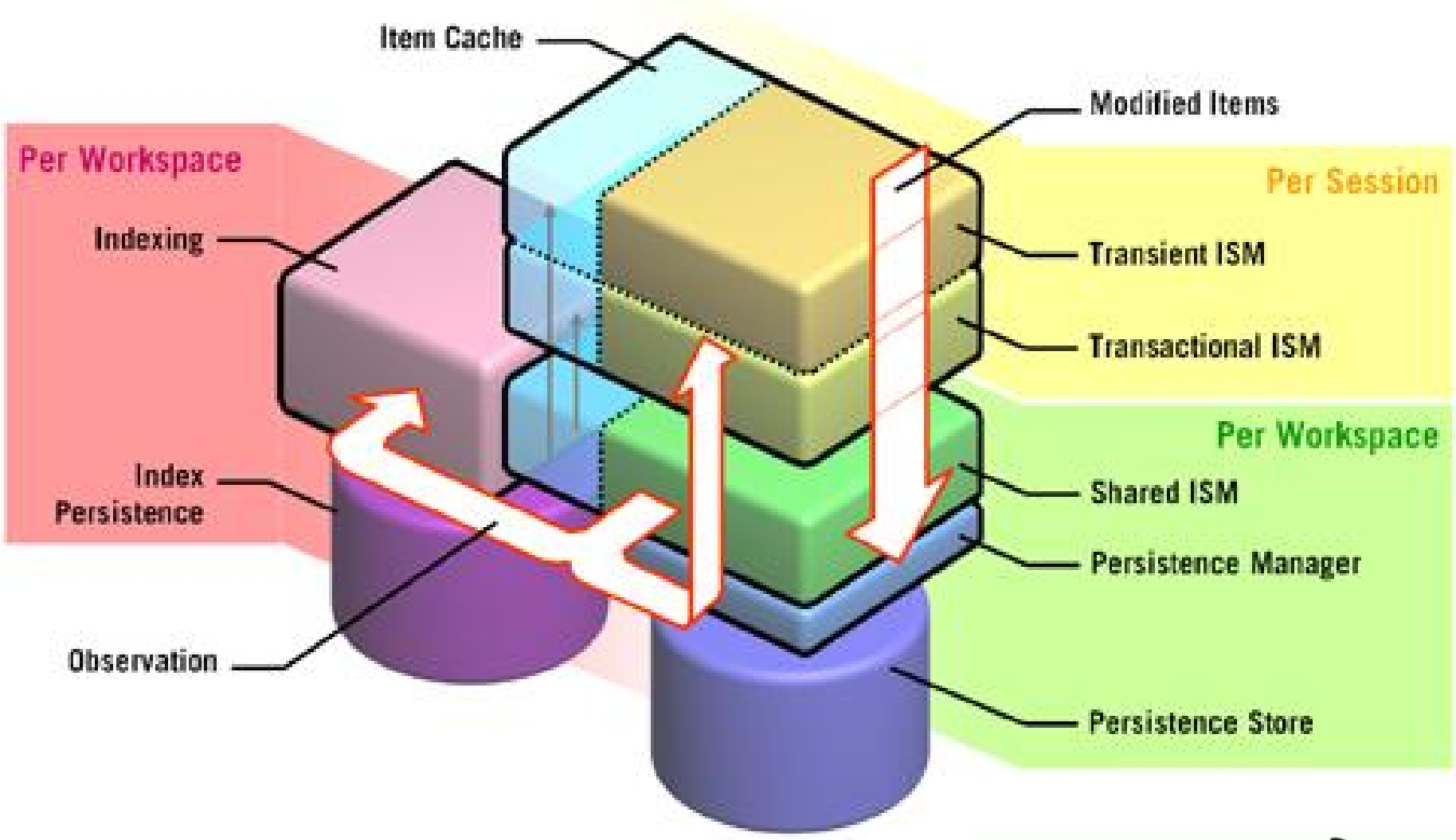
- mix:referenceable
- mix:lockable
- mix:versionable
- mix:created
- mix:lastModified
- ...

Apache Jackrabbit™

Internals - Search

- Workspace-managed JCR QueryManager
- Supported techniques:
  - JCR-SQL (JCR 1.0, now deprecated)
  - JCR-XPath (JCR 1.0, now deprecated)
  - JCR-SQL2 (JCR 2.0)
  - JCR-JQOM (JCR 2.0 / Java Query Object Model)
  - Direct referencing by UUID
- Backed by Apache Lucene
- Full-text search API
  - jcr:contains(), jcr:like(), wildcards
  - Monitor your search performance
  - Especially when combining jcr:like(), wildcards and deep hierarchies

Internals - Structural overview

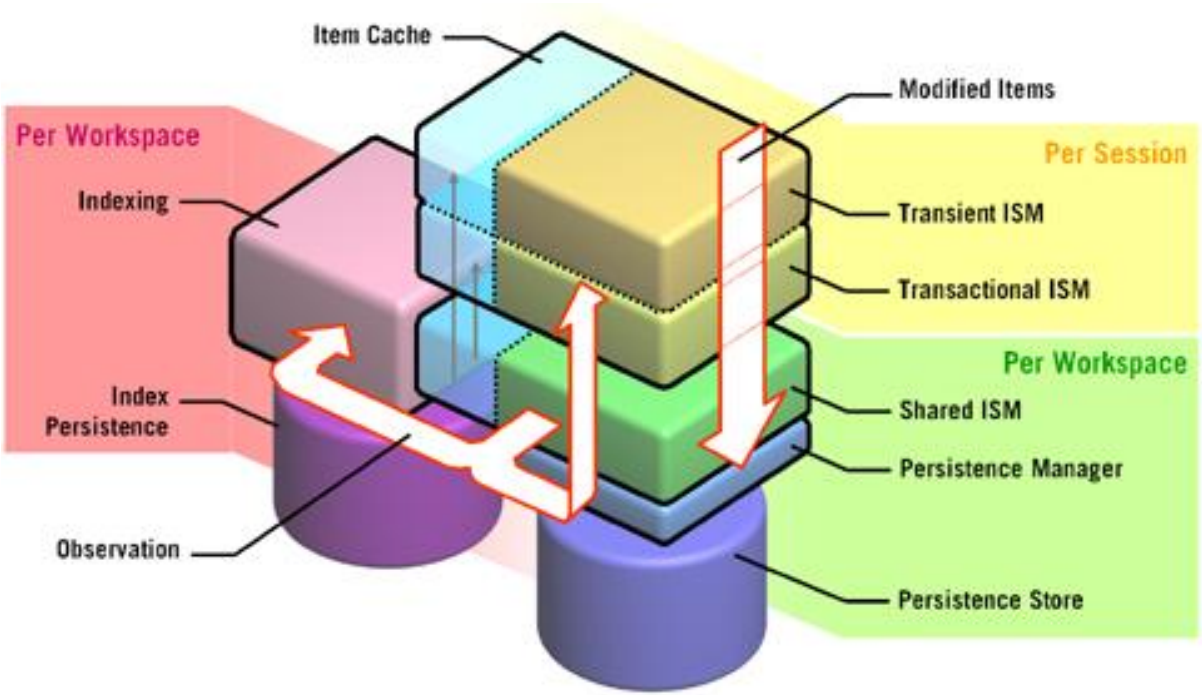


Apache Jackrabbit™

Internals - Statechanges (1)

Transient ISM

- Read cache for items which are read in the session
- Changes only visible in session (transient space)
- Other sessions won't see the change (yet)!

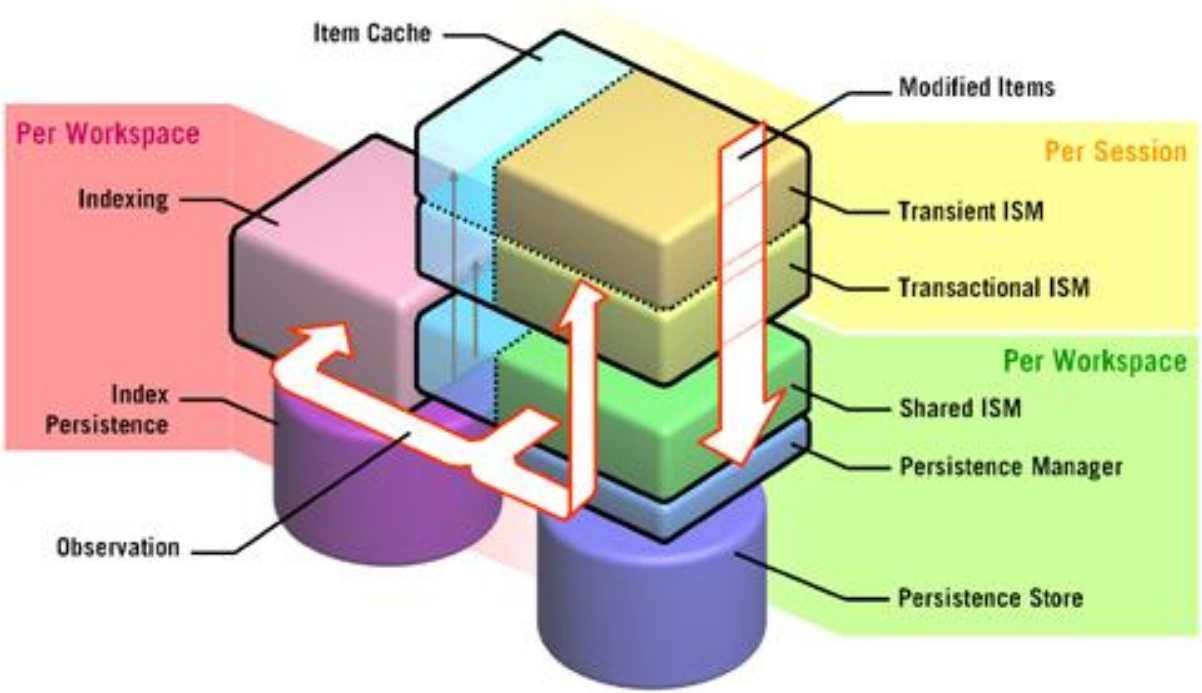


Apache Jackrabbit™

Internals - Statechanges (2)

Transactional ISM

- Read cache for items which are read in a transaction
- Promote item from transient to transactional state by calling .save()
- Changes only visible in session (transaction space)
- Other sessions will see the change after .commit()

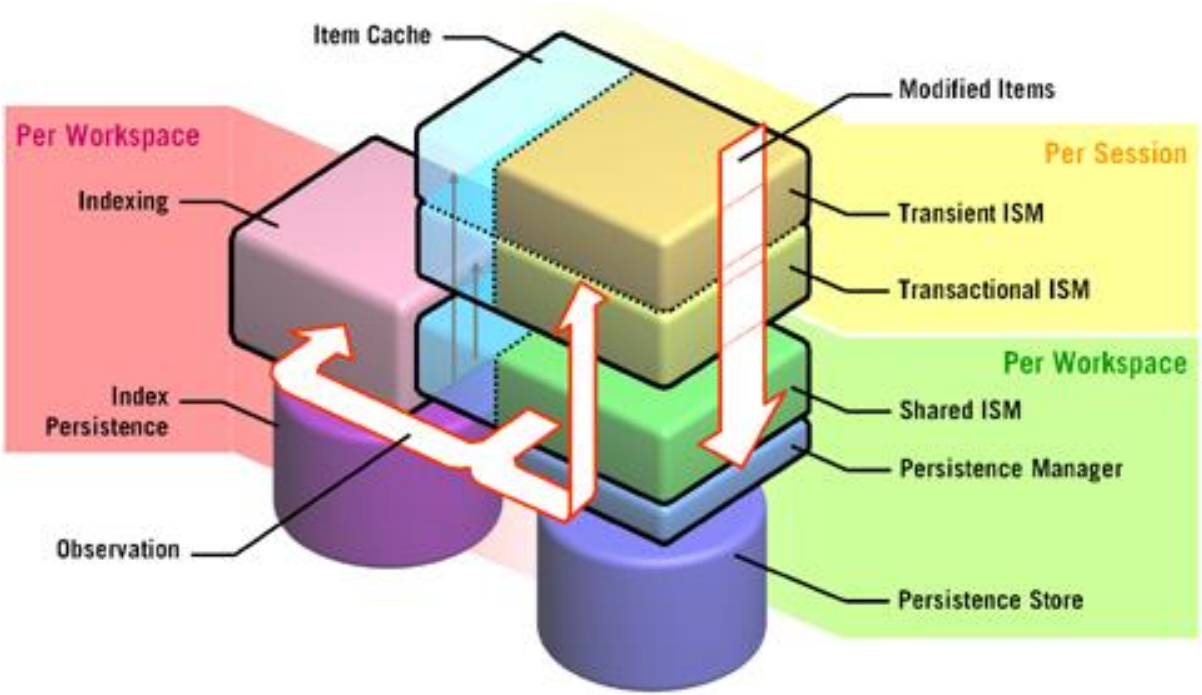


Apache Jackrabbit™

Internals - Statechanges (3)

Shared ISM

- Read cache for items which are read in the workspace
- Commit() sends a changelog to the Shared ISM
- Shared ISM propagates changelog to all sessions logged in to the same workspace and to the Persistence Manager. Also triggers observation.

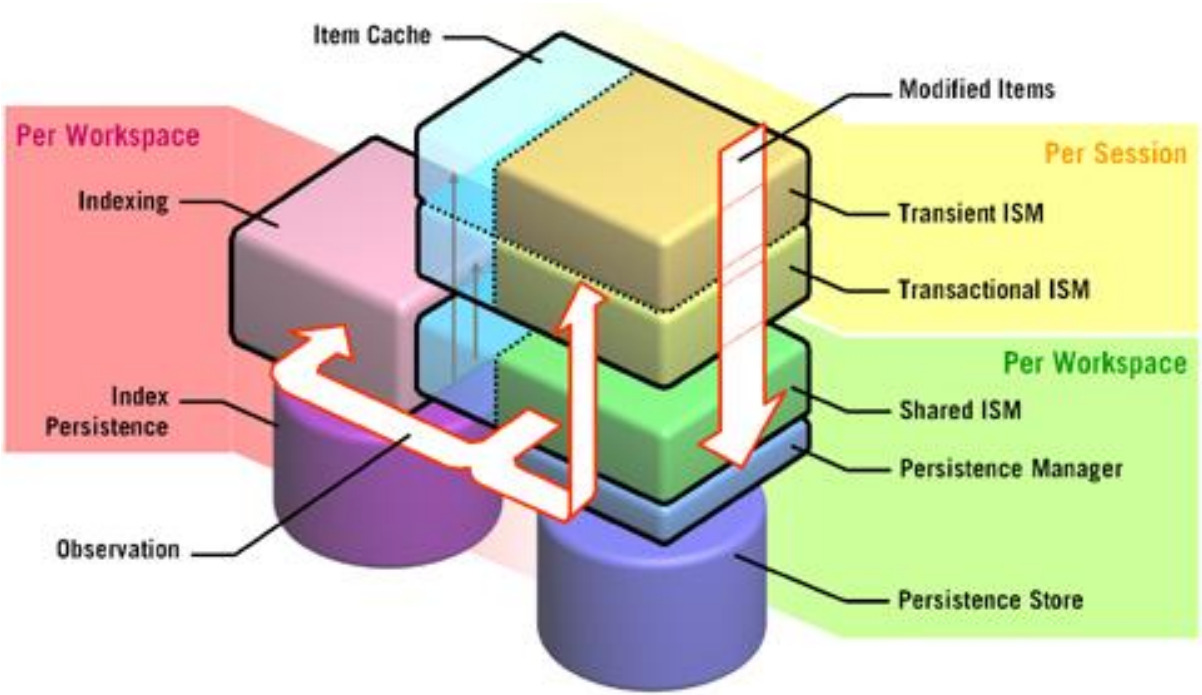


Apache Jackrabbit™

Internals - Statechanges (4)

Persistence Manager

- Persists Item States from the changelog
- Retrieval and storage of an item by it's item ID
- Bundle persistency, storing one item per bundle

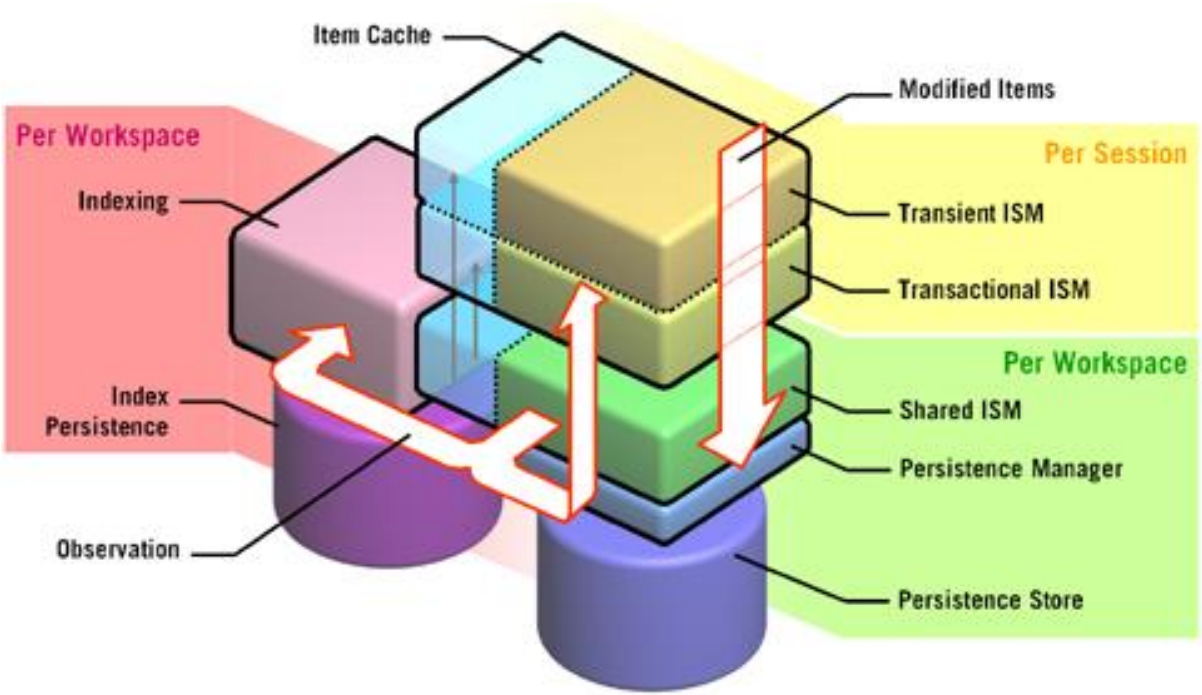


Apache Jackrabbit™

Internals - Statechanges (5)

Observation

- Triggered by the Shared ISM on commit()
- Allows applications to (a)synchronously subscribe to item state changes in the workspace

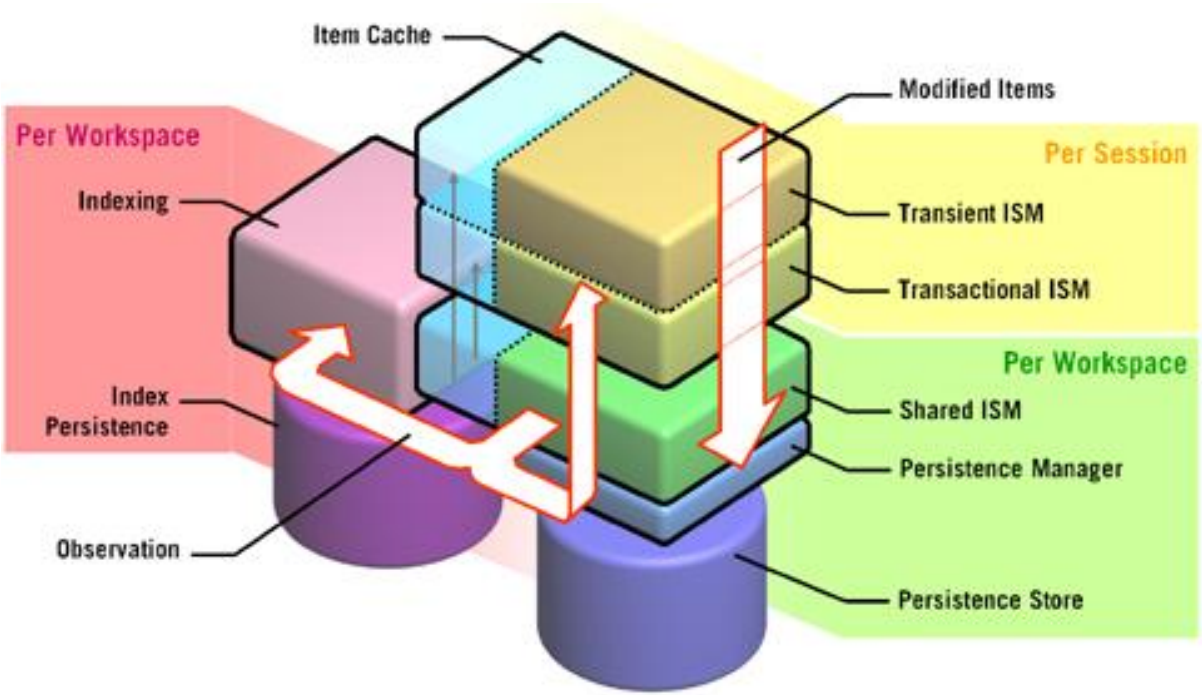


Apache Jackrabbit™

Internals - Statechanges (6)

Indexing

- Triggered by a synchronous observation event
- Instructs the Query Manager to index the modified state
- Facilitates complex retrieval (e.g. full text searches, hierarchical queries, etc.)



Apache Jackrabbit™

Apache Jackrabbit™

Internals - Locking

- Node-level locking by using "mix:lockable"
- Both shallow and deep locking
  - Shallow = self
  - Deep = self + children nodes
- Workspace-managed JCR LockManager
- Transactional lockdown:
  - Step 1: Get a lock in a separate transaction, since lockdown has to propagate via the Shared ISM
  - Step 2: Fire transactional operations requiring lockdown in a separate transaction
  - Step 3: Release lock before committing

Apache Jackrabbit™

Internals - Clustering (1)

- Cluster-wide
  - All cluster nodes need to access the **same** persistent storage. Single point of failure, failover required!
- Node-specific
  - (private) repository and workspace filesystem
  - search indices
- Modification on a node is reported to a **shared, cluster-wide journal**.
- All nodes read the journal every n msecs and update accordingly
- No race conditions by locking the update of journalised nodes

Internals - Clustering (2)

- CAP - Consistency
  - After change propagation by journal all nodes are consistent with each other -> not immediately, takes time to propagate, so only eventually consistent!
- CAP - Availability
  - After processing a journal update, the node is indexed and thus calls are serviced.
  - Before or while processing a journal update, the node is not indexed, so calls are not serviced, throwing exceptions (so, no wait-state observer).
- CAP - Partition tolerance
  - On message loss or system failure, the journal can be replayed to reconstruct node state.

Framework support

- Spring:
  - Spring Extensions JCR
    Unofficial, not incorporated in Spring Modules (http://se-jcr.sourceforge.net)
- Apache Jackrabbit OCM
  - Official annotation-based abstraction on javax.jcr.*
  - Preferred way for production due to stability
    http://jackrabbit.apache.org/object-content-mapping.html
- DIY based on javax.jcr
  - No abstraction = better understanding at internals
  - That's why I used javax.jcr in the hands-on :-)!

Apache Jackrabbit™

HackRabbit Hands-On

- Free-form (Spring, OCM, javax.jcr, ...)
- Assignment 1 - Fix the bootstrappertests
  - Gain a **very basic** understanding on repository structure and storage
  - Focus on nt:unstructured instead of namespaced nodetypes, since NoSQL is about 'programmer freedom'
- Assignment 2 - Fix the querytests
  - Gain a **very basic** understanding on repository data retrieval
  - By JCR-SQL and JCR-XPath; deprecated, but has least required knowledge