

**adaptTo()**

APACHE SLING & FRIENDS TECH MEETUP  
BERLIN, 26-28 SEPTEMBER 2012

**APACHE JACKRABBIT: BASIC CONCEPTS**

Christian Riemath, Igor Sechyn

## Igor Sechyn

Senior CMS Developer

## Christian Riemath

Senior CMS Developer

### pro!vision GmbH

Löwengasse 27A

60385 Frankfurt

<http://www.pro-vision.de>



**PRO!VISION**  
SOFTWARE CRAFTSMANSHIP



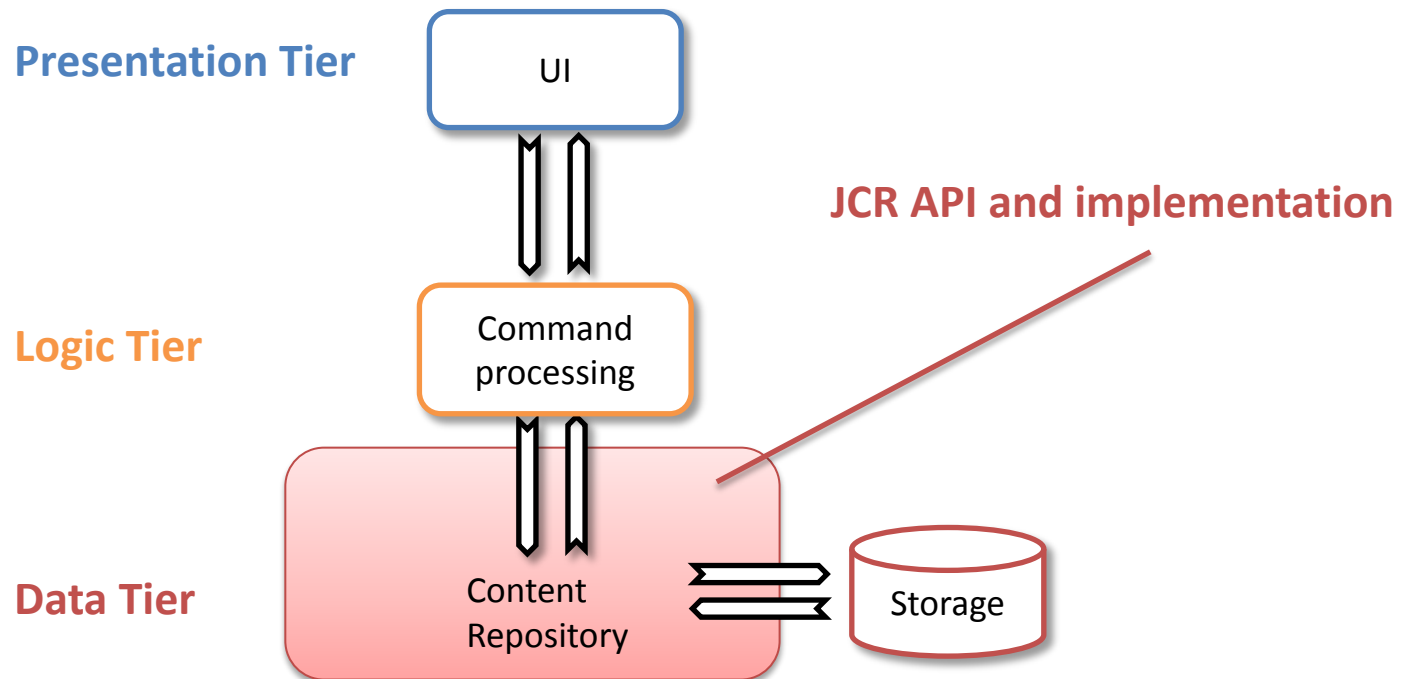
Spezialized on Adobe CQ and it's technology stack since 2003

# Agenda

- Introduction
- What is Content?
- Content Repositories
- JCR vs. RDBMS
- JCR API Objectives
- JCR Artifacts
- Apache Jackrabbit
- Code Examples

# Introduction – The big picture

- It's about the data layer
- It's about a special type of data: content



# What is content?

- The JCR mantra: Everything is content
  - Application domain specific content (Articles, blog posts, comments, assets, ...)
  - Structured data (e.g. an address record in a database)
  - Unstructured / Inherently structured data (binary documents like invoices, product description etc.)
  - Workflow definitions
  - Access Control Lists
  - Code
  - etc.

# What is content?

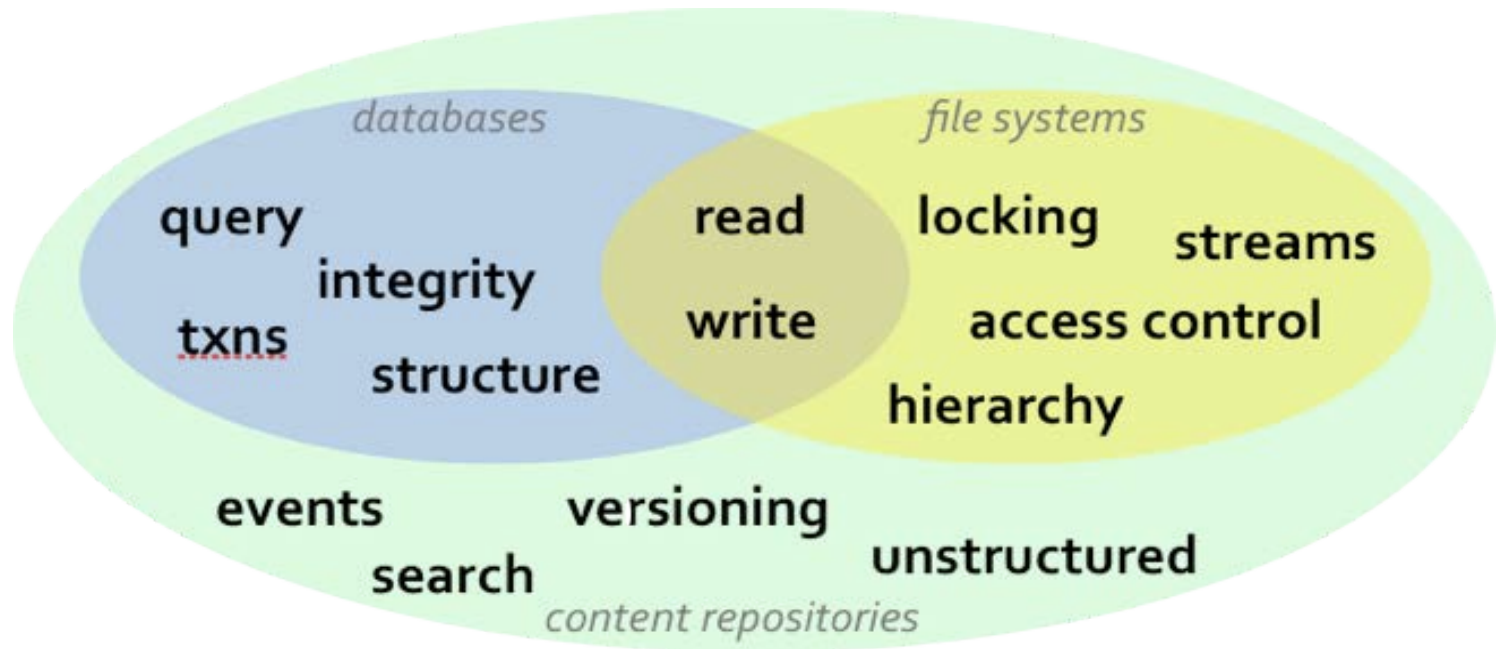
- Attributes that distinguish *content* from *data*
  - Shall provide information for an audience (=> finding and accessing this information should be easy)
  - Is always embedded inside a (hierarchical) context (=> this supports accessing and finding information)
  - Created and edited by (different) authors (=> locking and versioning necessary)
  - Managed within a publishing workflow (=> eventing functionality has to be applicable)

# Content repository (1)

- Single repository for all types of content
- Support (multi-)hierarchical structure (=> *Findability*)
- Provide flexibility: Permit and simplify change of structure and metadata
- Assist query and search
- Provide access control
- Provide state control (by lock and version management and indicating the publication state)
- Assure data integrity

# Content repository (2)

- Combines attributes of file systems and databases and adds other features





# JCR vs. RDBMS

	JCR	RDBMS
<b>Structure</b>	Unstructured, semi-unstructured, structured	Structured
<b>Navigation</b>	Navigation API, Traversal access, direct access, write access	Not supported
<b>Access control</b>	Record level	Table and column level
<b>Version control</b>	Supported	Not supported
<b>Code complexity</b>	Simple for navigation Complex for operations	Complex for navigation Simple for operations
<b>Changeability</b>	More agile Decoupled from the application	More rigid Coupled with the application

# JCR vs. RDBMS

- It is
  - NOT an issue of better or worse
  - but of what best suits the scenario and requirements
- JCR
  - does not replace the RDBMS
  - provides an alternative data model for specific requirements encountered in content management and collaborative applications
- Requirements and best possible fit for problem solving should dictate the choice of technology

# JCR API Specification Objectives

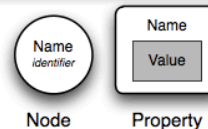
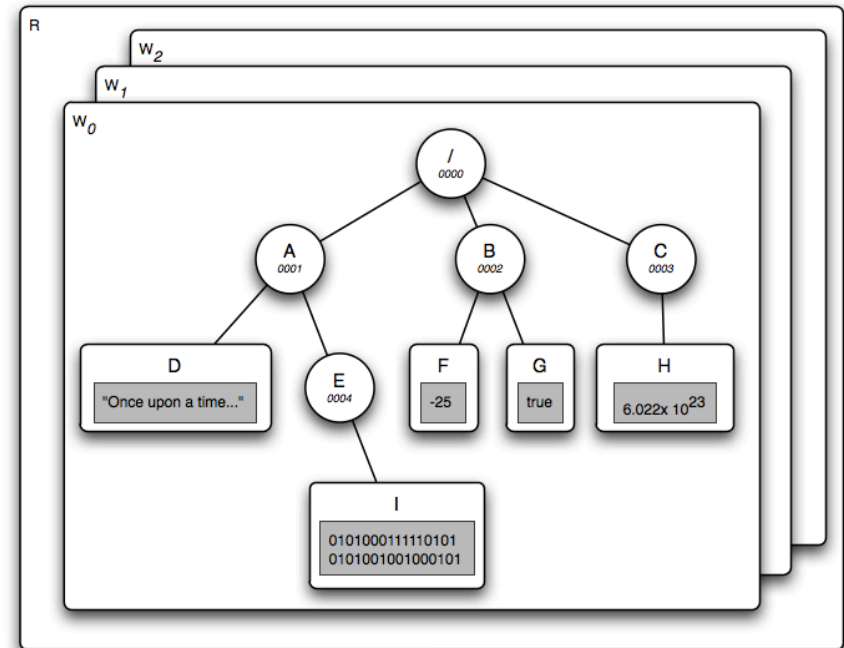
- Provide API to access content repositories in a uniform manner
- Abstract where and how the content is stored
- Leave the dirty details for handling the underlying storage system (file system, RDBMS etc.) to the API implementation vendors
- ... but allow for relatively easy implementation on top of a wide variety of existing content repositories as possible

# JCR – The Standards

- JCR 1.0 (JSR-170; final version released in 2005):
  - Level 1: defines a read-only repository, including introspection of content-type definitions, export of content to XML and searching
  - Level 2: writing content, assignment of types to content, reference tracking and integrity and importing content from XML
- JCR.2.0 (JSR-283; final version released in 2009):
  - Query extensions: Abstract Query Model, Java Query Object Model, SQL (Xpath queries deprecated)
  - Shareable nodes
  - All features implemented in Apache Jackrabbit (August 2009)
- JCR 2.1 (JSR-333; not final yet):
  - Small API improvements
  - PHPCR

# The JCR repository model (1)

- **Repository**
  - consists of one or more workspaces
  - provides access to the workspaces through sessions
- **Workspace**
  - is a directed acyclic graph of items
  - is identified by a unique name
  - holds at least one item, which is a root node



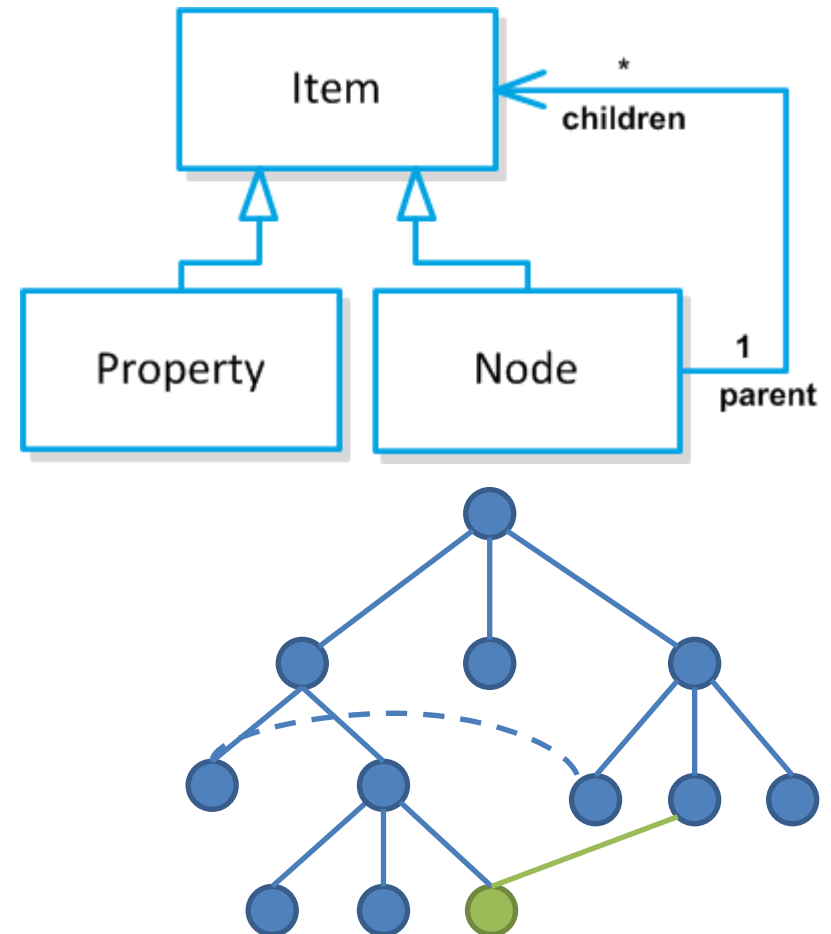
- R = Repository
- W<sub>0..2</sub> = Workspaces

# The JCR repository model (2)

- **Session**
  - Binds the user to the exactly one requested Workspace
  - Workspace can be bound to multiple sessions
  - Defines the level of authorization based on the users credentials
  - Provides CRUD operations on the nodes of the bound workspace

# The JCR repository model (3)

- **Item**
  - is an abstraction level model
  - can be a node or a property
- **Nodes**
  - form the structure of the data
  - may have zero or more properties
  - always have a parent node, except for the root node
  - always have one primary node type
  - may have zero or more mixin types
- **Properties**
  - hold the actual data
  - can be single or multi-valued
  - have one of 12 possible types (string, boolean, data, reference, etc.)



# The JCR repository model (4) – Node types

- **Node types**
  - enforce structural restrictions on the node and its properties
  - can be declared as mixins, which is a boolean flag on the node type definition
- **Primary node types**
  - define the core characteristics of a node
- **Mixin types**
  - add additional characteristics, related to specific functions or metadata



# The JCR repository model (5) – Example node

contact

jcr:primaryType	nt:unstructured
name	Max Mustermann
position	Manager
department	Human Resources
jcr:created	2014-03-11T11:03:11

# Versioning

- History of content's changes
- Version – saved state of content
- The process of saving a certain state of content is called “checkin”
  - The node has a mixin “mix:versionable”
  - The node was previously “checked out”
- Version of a node includes the node's children

# Observation

- Notification of persistent changes to a workspace
- Asynchronous observation
  - respond to changes made in a workspace as they occur
- Journalled observation
  - report of changes that have occurred since some specified point in the past
- Event driven
  - NODE\_ADDED
  - NODE\_MOVED
  - NODE\_REMOVED
  - PROPERTY\_ADDED
  - PROPERTY\_REMOVED
  - PROPERTY\_CHANGED

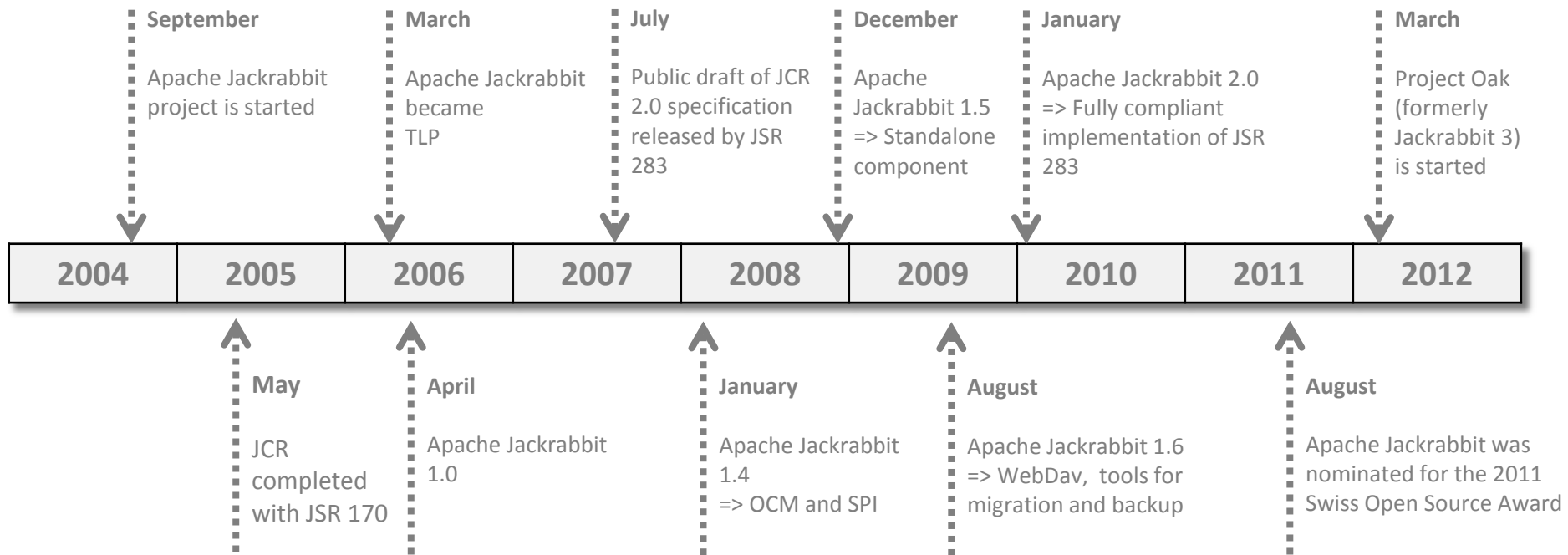
- **Access Control Management**
  - Privilege discovery
  - Assigning access control policies
- **Privilege**
  - Set of operations on a node
  - JCR API defines standard privileges
- **Access Control Policies**
  - Grant privileges to users
  - Semantics are implementation specific

- Query for content that meets user-defined criteria
- JCR 2.0 enables support for multiple query languages
  - specifies JCR-SQL2 and JCR-QOM
- JCR 1.0 specified
  - XPATH
  - JCR-SQL
- Provides Query API to perform all query operations

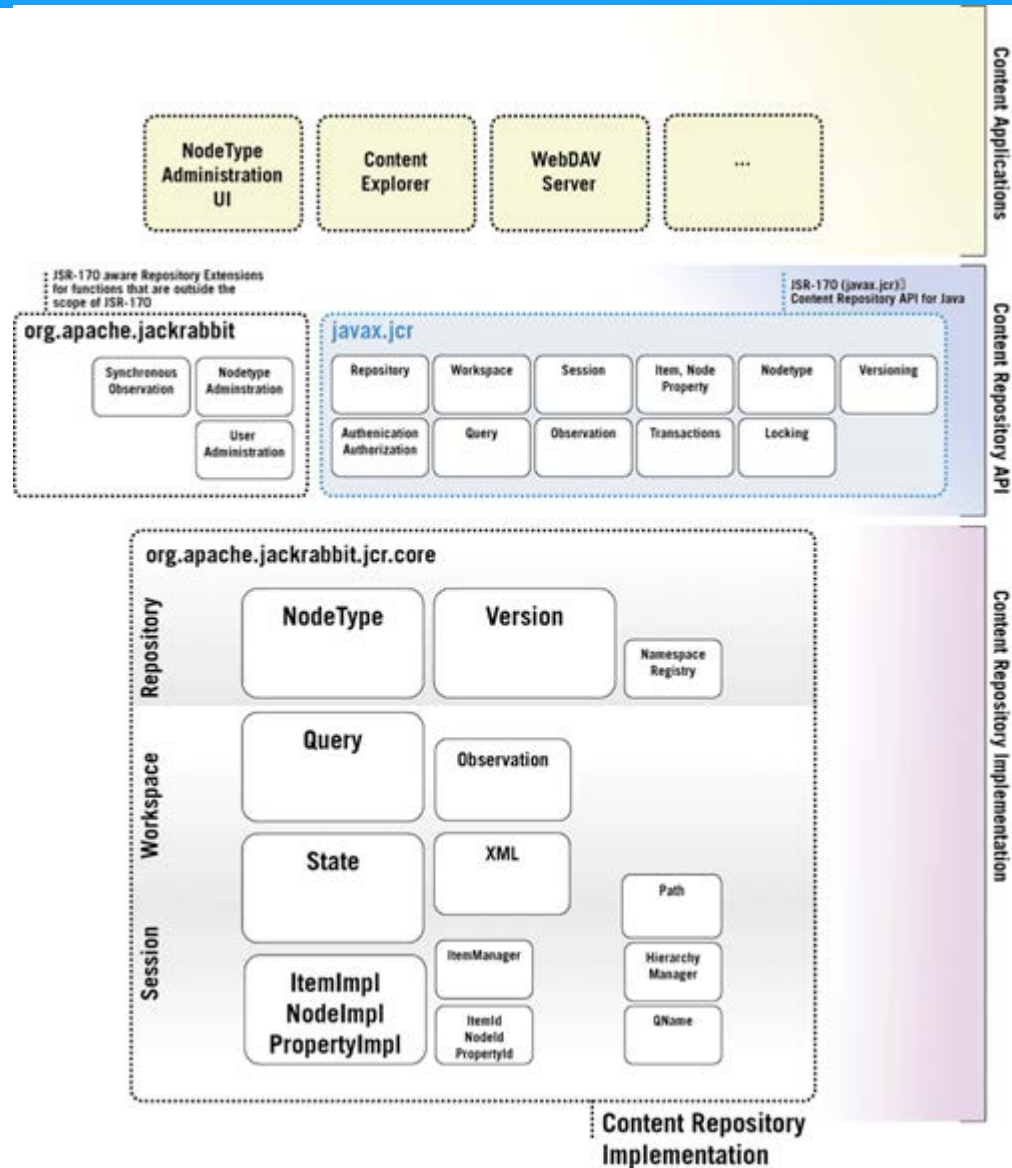
# What is Apache Jackrabbit

- Top-level project of the Apache Software Foundation
- Current release: Apache Jackrabbit 2.5 (June 2<sup>nd</sup>, 2012)
  - Jackrabbit OAK
- Open Source Content Repository
- Supports storing, accessing and managing content
- Supports structured and unstructured content
- Fulfills both level of JCR compliance
- Reference Implementation of JCR
- <http://jackrabbit.apache.org/>

# Timeline



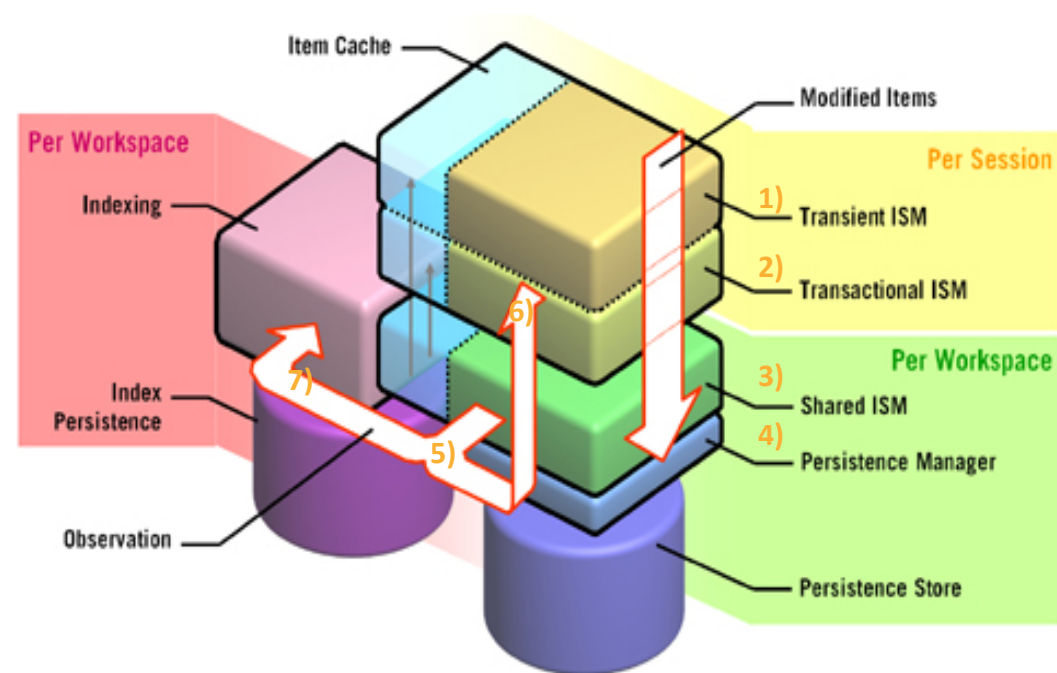
# Jackrabbit architecture (1)

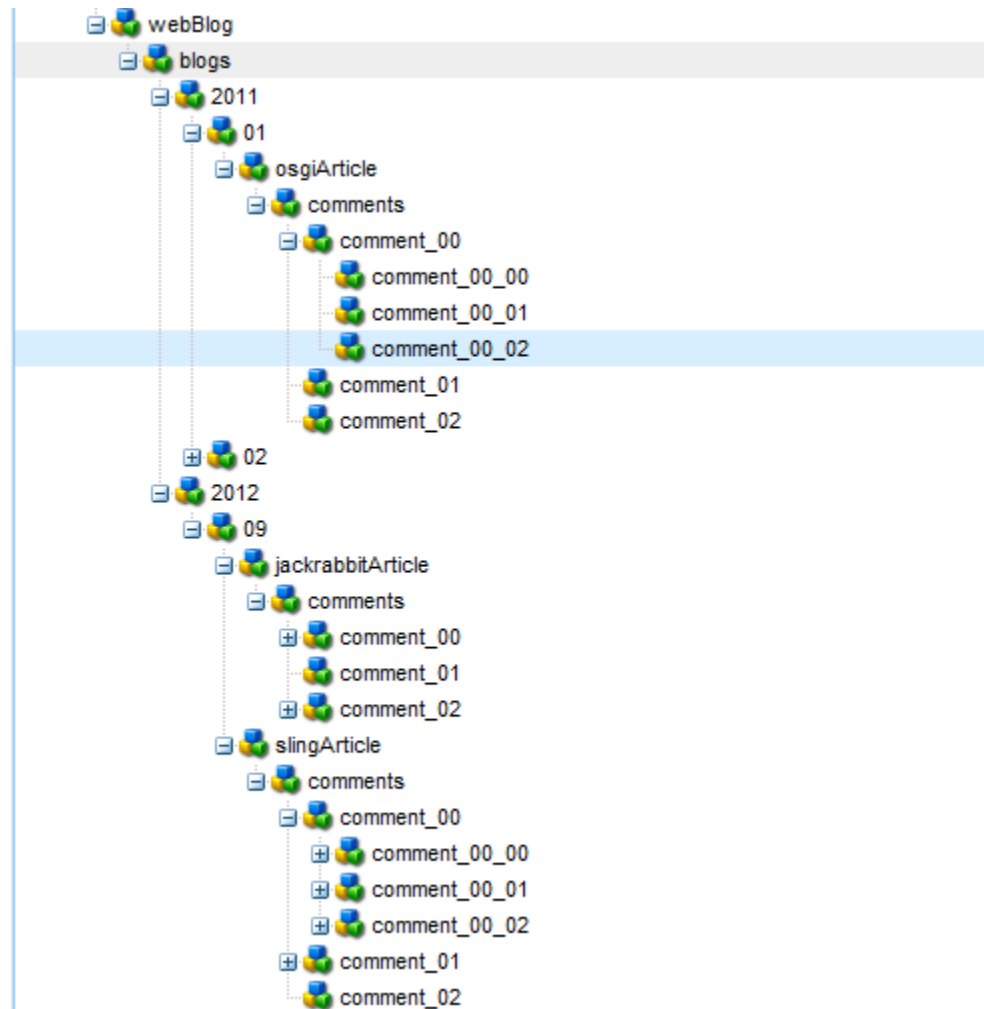


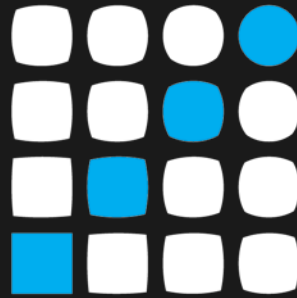


# Jackrabbit architecture (2)

1. Modified items cached
2. Saved items promoted
3. Committed items notified
4. Committed items persisted
5. Observation triggered
6. Applications subscribe changes
7. New/Modified Items indexed







**adaptTo()**

APACHE SLING & FRIENDS TECH MEETUP  
BERLIN, 26-28 SEPTEMBER 2012

**Thank you for your attention!**

**Any questions?**