Integrating CMS/DMS into OpenOffice

Michael E. Bohn
Consultant Office Migration
Sun Microsystems GmbH

Agenda

- Motivation
- CMS/DMS integration – current approach.
- What is the Java Content Repository.
- Snapshot of the current implementation.
- Demo.

Motivation

- Growing popularity of the Open Document Format.
- Growing demand to manage information.
- Increasing number of proprietary CMS/DMS.
- Providing a common extension for all CMS/DMS.
- Simplification of accessing and sharing informations.
- Simplification of the administration of office installation.
- Making OpenOffice.org available for more business solutions.
- Making a migration much easier.

CMS/DMS integration – current approach

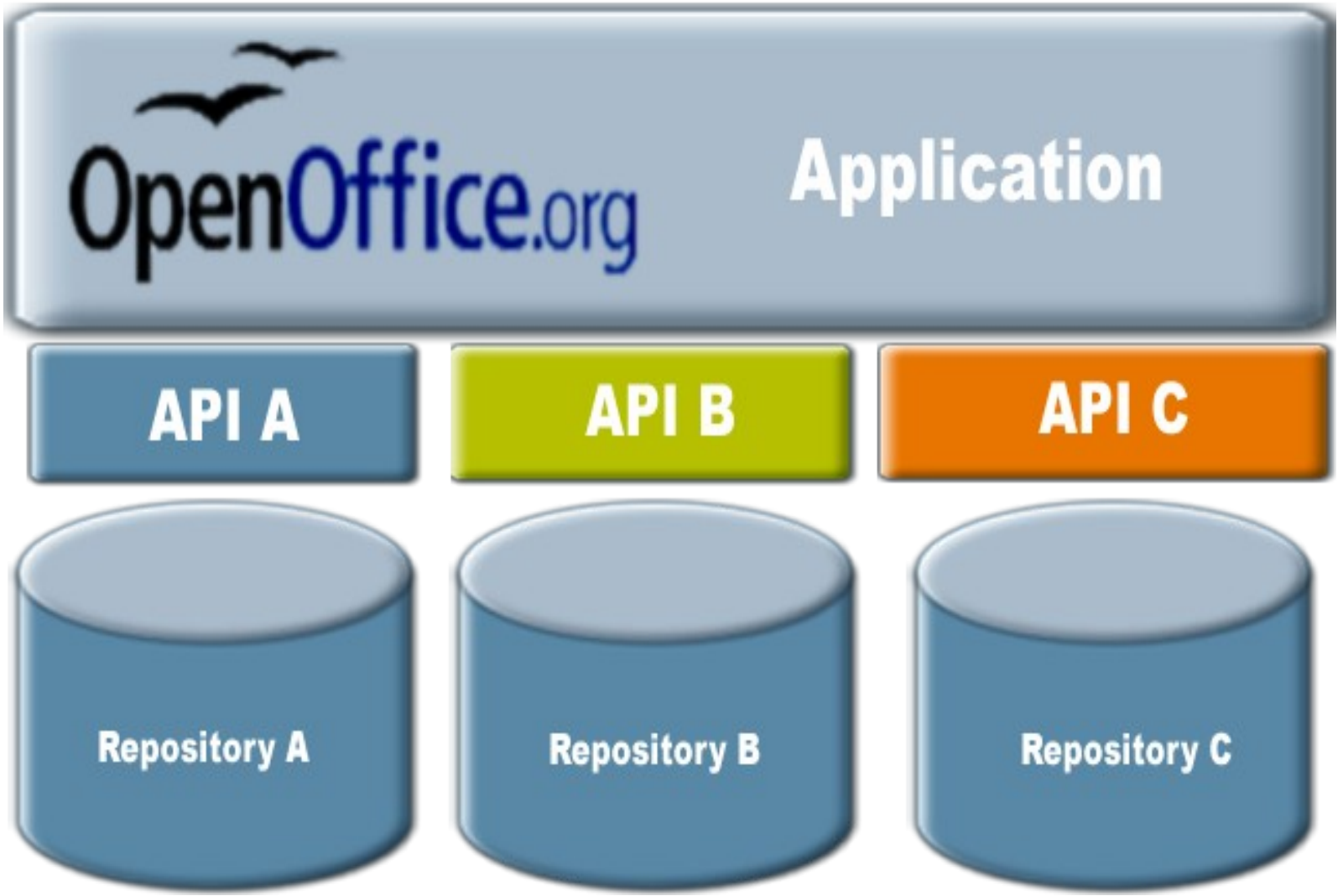
CMS/DMS integration

- Different DMS provide propriety integration
  - Alfresco
  - FileNet
  - OpenText
- Each DMS uses his own API for communication
  - Alfesco
    - HTTP
  - FileNet
    - HTTP and Web-DAV
  - OpenText
    - .Net component

CMS/DMS integration – features provided

- Common
  - Browsing folders
  - Checking files out and in
  - Opening and saving documents
  - Searching for documents
  - Attaching keywords to documents
- Special
  - Managing work flows
  - Converting documents

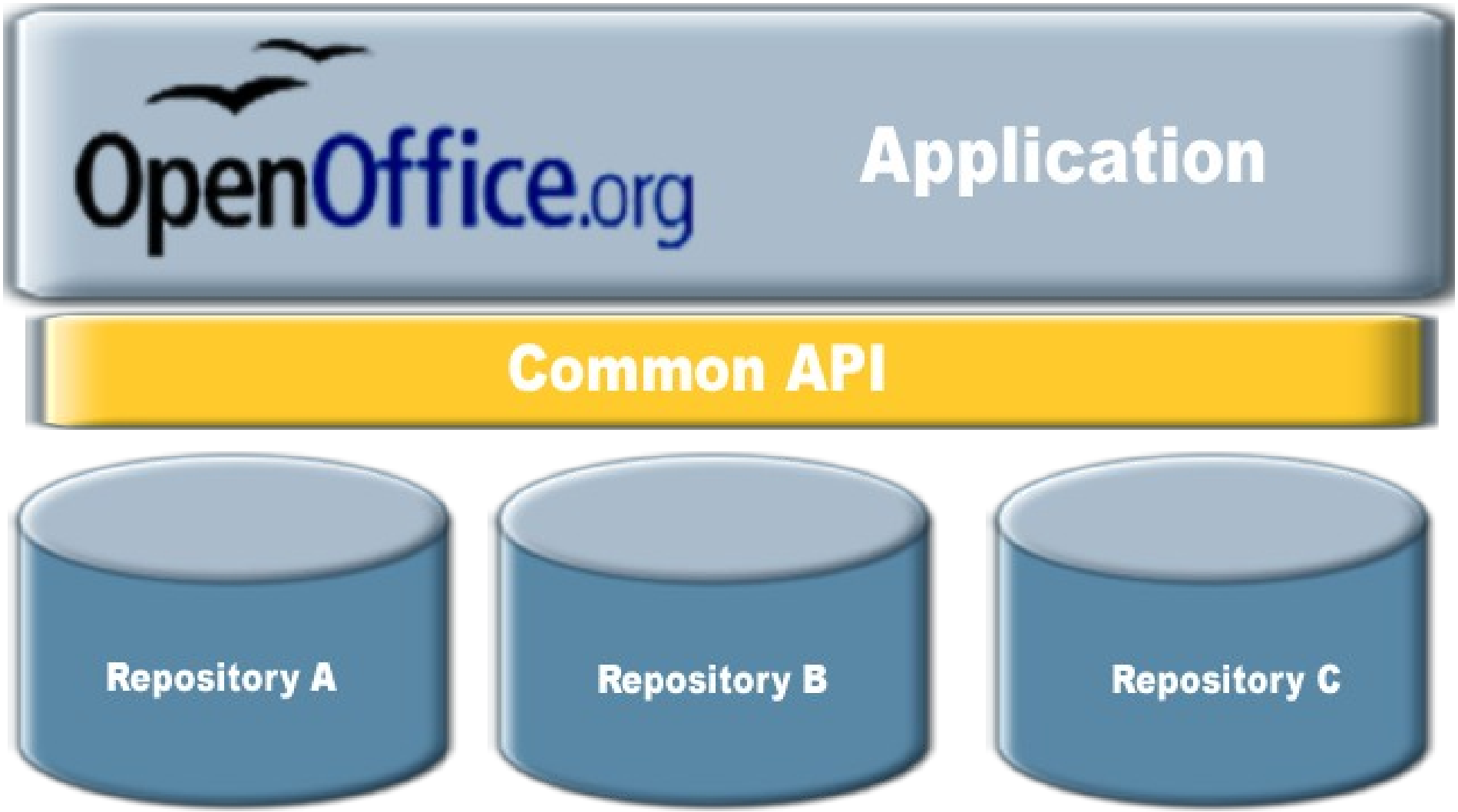
CMS/DMS integration – current approach



CMS/DMS integration – current approach

- Many communication interfaces = Many extensions.
- High learning curve for the user.
- Complicated administration of the office installation.
- Developer need to know each system.
- User can not exchange their underlaying DMS.
- Migration more complicated.

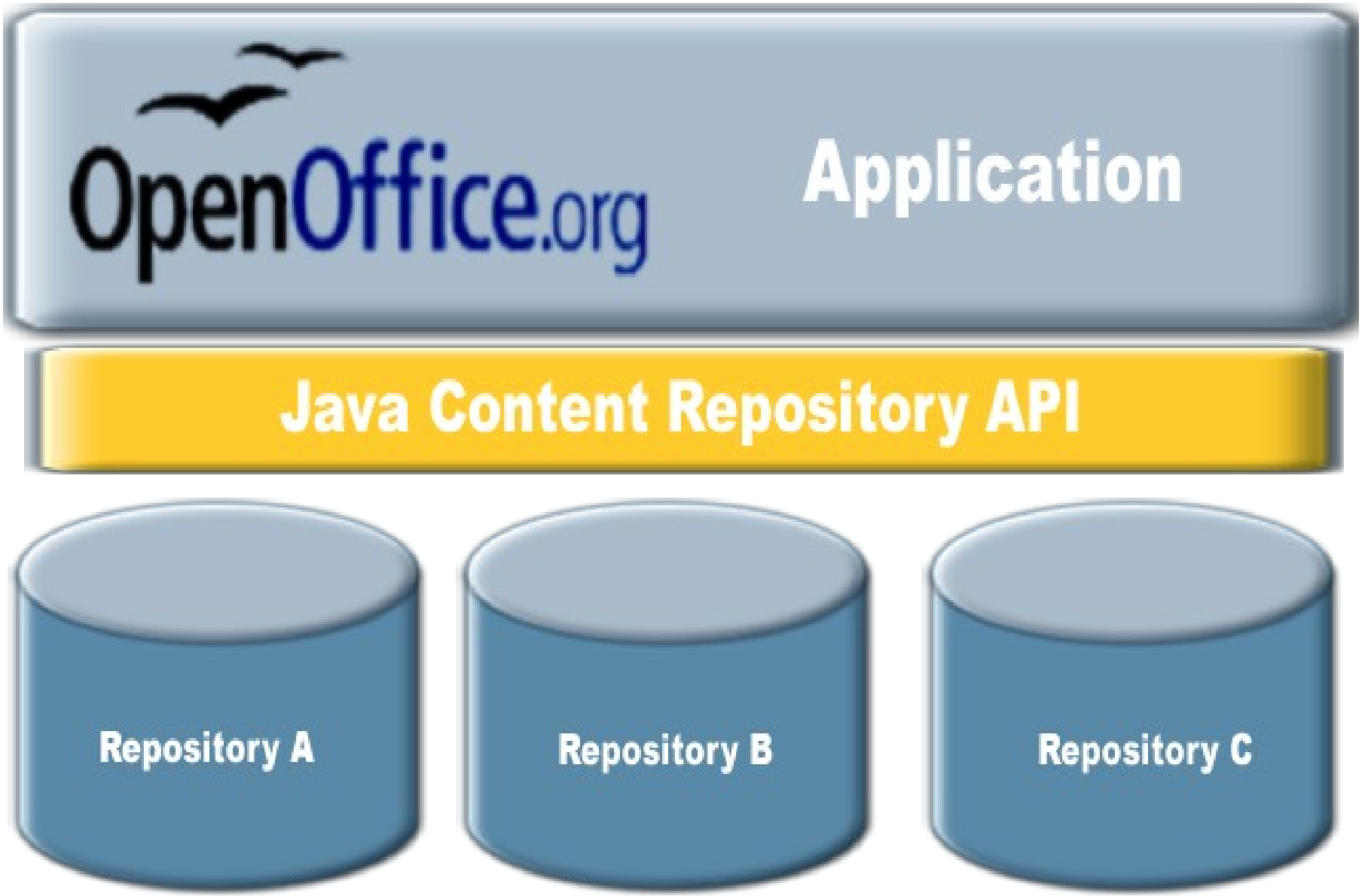
CMS/DMS integration – new approach



CMS/DMS integration – new approach

- One communication interfaces = one extension.
- User just needs to know one extension.
- Simple administration of the office installation.
- Developer can extend the function easily.
- Customers can exchange their underlaying DMS.
- Migration much easier.

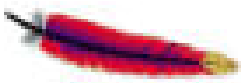
CMS/DMS integration – using JCR



Why using JCR as the common interface?

- JCR Officially released June 17, 2005.
- Two levels of compatibility with optional features.
- JCR is widely adopted.
- Many CM- an DM systems support an JCR API
- Can be used locally and over the INTERNET

Systems supporting JCR



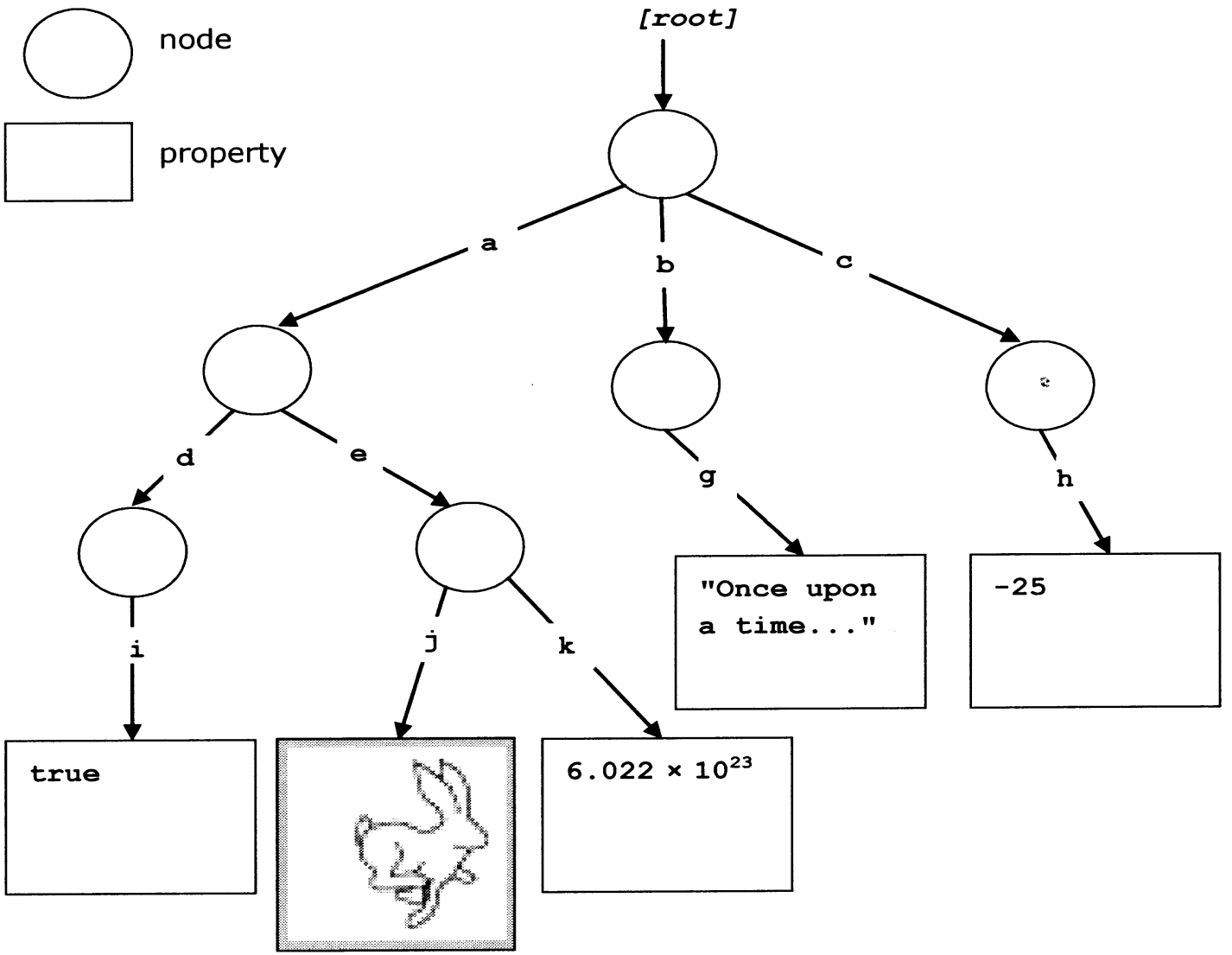
Many, many, many more……

What is Java Content Repository

What is Java Content Repository.

- Consists of workspaces.
- Workspace contains a tree of items.
- An item can either be a node or a property.
- Properties can only be a leaf
- Nodes encapsulate the content structure
- The actual content of the repository is stored in the values of the properties.
- Level 1 – API functions to read the repository.
- Level 2 – API functions to write into the repository.
- XML and SQL queries supported

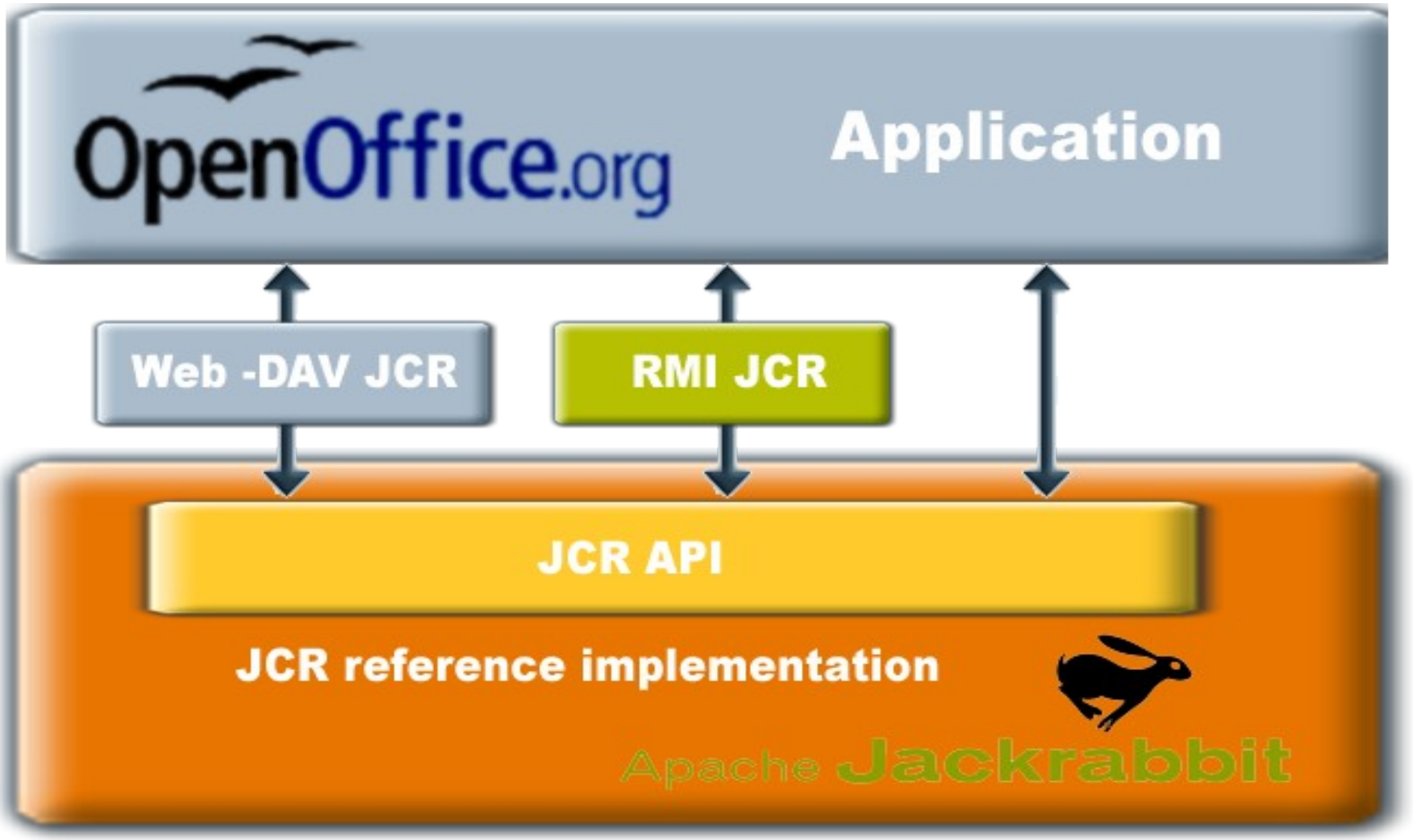
What is the Java Content Repository



How to use JCR with OpenOffice.org

- Jackrabbit is a reference implementation for JCR.
- Communication ways provided by Jackrabbit.
  - Web-DAV - JCR API methods called via Web-DAV
  - Remote API – functions called by using RMI
  - Local API – functions called directly
- Problems
  - JCR with Web – DAV is not provided by all systems
  - RMI is provided differently- difficult to configure on the client
  - Local API is called differently on each DMS

JCR - Communication ways



Getting JCR repository.

```
// Setup Spring and Transaction Service
ApplicationContext context = new ClassPathXmlApplicationContext("classpath:alfresco/application-context.xml");

// Retrieve Repository
Repository repository = (Repository)context.getBean("JCR.Repository");

// Login to workspace
// Note: Default workspace is the one used by Alfresco Web Client which contains all the Spaces
//       and their documents
Session session = repository.login(new SimpleCredentials("admin", "admin".toCharArray()));
```

```
import javax.jcr.Repository;
import javax.servlet.ServletContext;

ServletContext context = ...; // context of your servlet
ServletContext jackrabbit =
    context.getContext("/jackrabbit-webapp-1.4");
Repository repository = (Repository)
    context.getAttribute(Repository.class.getName()).
```

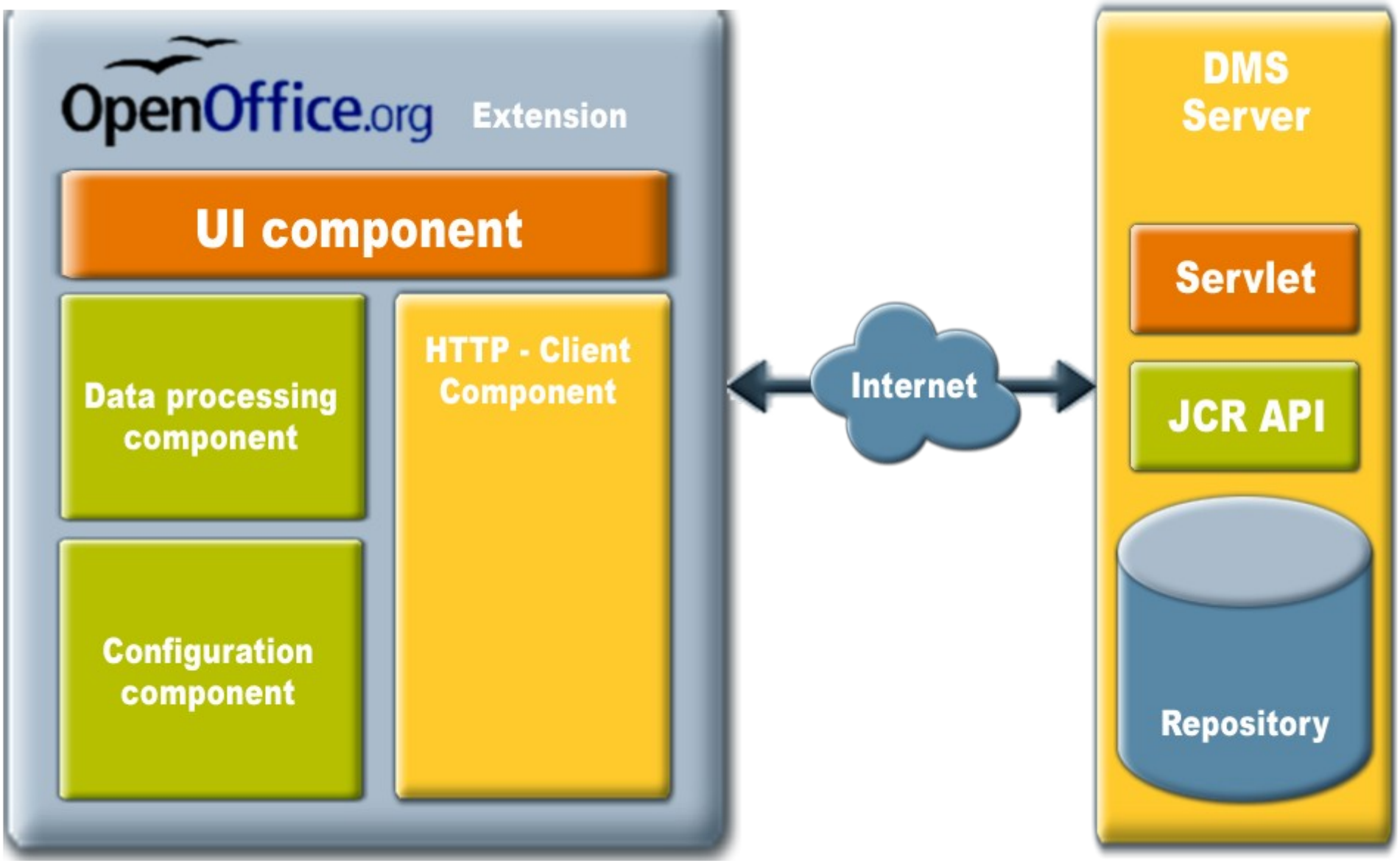
Current implementation

- Supposed as prof of concept
- Communication
  - Set of actions has been defined
  - XML is used to transport data between client and server
- Serve side
  - Separate Web application
  - Servlet that acquires the repository object
- Client side
  - Java extension
  - HTTP-Client used for communicating with the server
  - XML – DOM used to analyze server responses

Current implementation

- Functions
  - Showing folders
  - Sowing a list of documents available in a folder
  - Creating folders
  - Loading documents
  - Saving documents
  - Checking documents in and out
- Servlet
  - supposed to be the template for DMS vendors

Snapshot of the current implementation



DEMO

Summary

- Increasing number of proprietary DMS system
  - > Many extensions
  - > complicated
- Common Interface
  - > Only one extension for all CMS/DMS
  - > Easier to maintain
  - > Easier to administrate
- Java Content Repository is used
  - > Standard and widely adopted
- Implementation
  - > Apache Jackrabbit
  - > Servlet

Q & A

Meet the Sun Experts at the Sun Booth.