Formación Alfresco

keensoft

Día 2 - Integración

Angel Borroy

Miembro fundador de Order of the Bee (http://orderofthebee.org))

- Programme Chair de la BeeCon2016 (http://beecon.buzz)
- Participante en el Comité de catalogación de addons

Ponente en conferencias internacionales

- Alfresco Summit 2013 (Barcelona)
- Alfresco Summit 2014 (London)
- BeeCon 2016 (Bruselas)
- Tech TalkLive #82 (https://www.youtube.com/watch?v=fy-dE9uOL-Y)
- Tech TalkLive #85 (https://www.youtube.com/watch?v=qz2LoVCU8Go)

Autor de diferentes contribuciones y addons para la plataforma

- GitHub (http://github.com/keensoft)

Angel Borroy

angel.borroy@keensoft.es

@AngelBorroy

GitHub

- http://github.com/keensoft
- http://github.com/angelborroy

Comunidad oficial de Alfresco

- http://community.alfresco.com

Blogs

- http://orderofthebee.org
- http://angelborroy.wordpress.com
- http://www.keensoft.es/blog

Día 2 – Integración

CMIS

- Apache Workbench
- curl
- Cliente Java
- Otros clientes CMIS: PHP, Python, .NET, Objective-C, JavaScript

REST API

- Alfresco Web Scripts
- OpenAPIs / swagger.io

Aikau

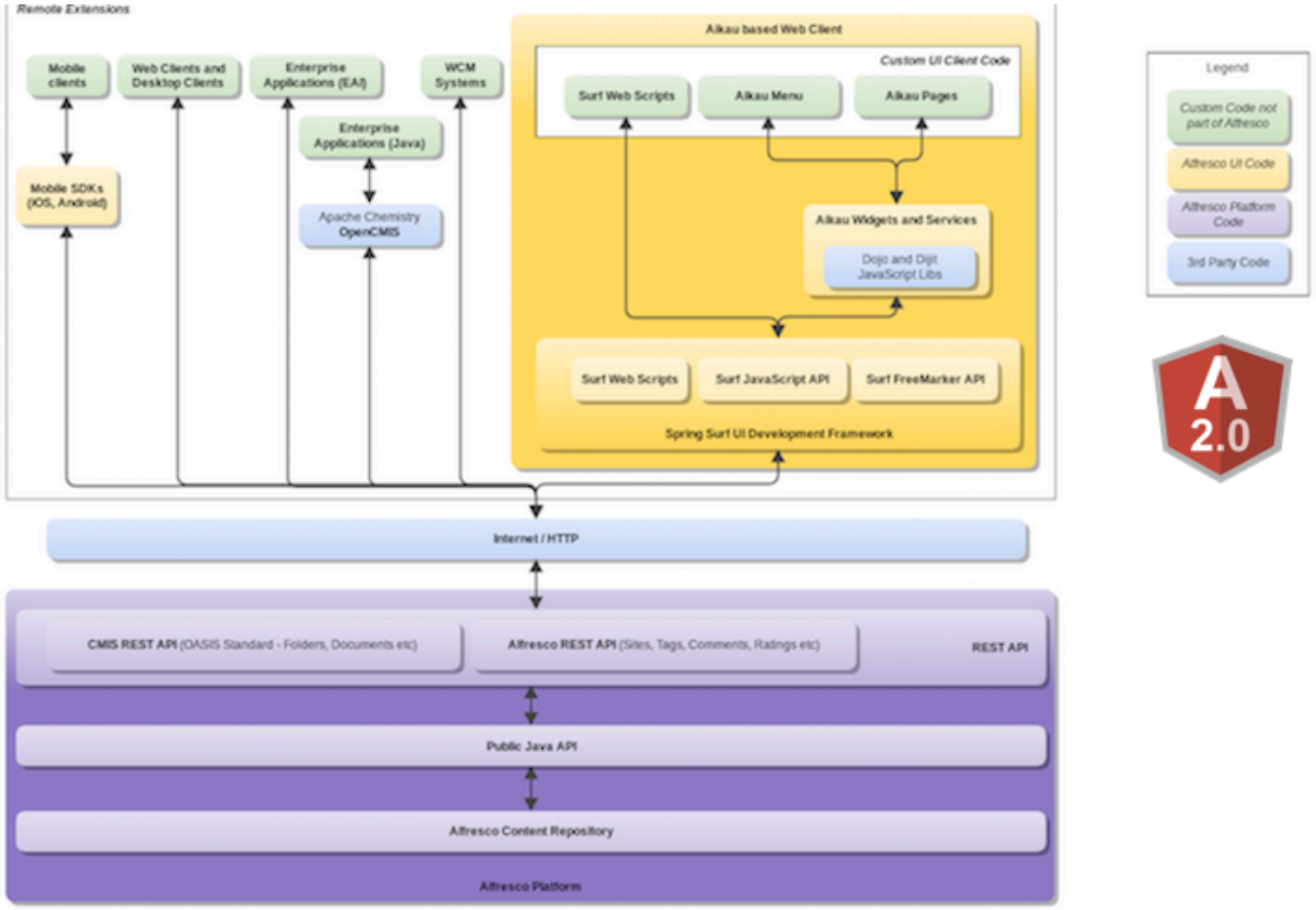
- Aplicación cliente

Alfresco Development Framework (ADF)

- Alfresco JavaScript Library for Node.js

Mecanismos de integración

Mecanismos de integración



CMIS

CMIS

Content Management Interoperability Services estándar OASIS que permite el acceso a repositorios de contenido

- Servidores ECM: Alfresco, ECM Documentum, HP Interwoven, IBM Content Manager y FileNet, Lotus Quickr, Microsoft SharePoint, OpenText, SAP
- Servidores WCM: Magnolia, Liferay, Drupal, Hippo, TYPE3, OpenCMS, docCMS
- Blogs: Wordpress
- Clientes: LibreOffice, Adobe Drive, Atlassian Confluence, SAP ECM Integration, Pentaho Data Integration, SugarCRM, Trac, Kofax,...
- SOA: Mule ESB, Spring Integration
- Librerías: Apache Chemistry (Java, Python, PHP, .NET, Objective-C)

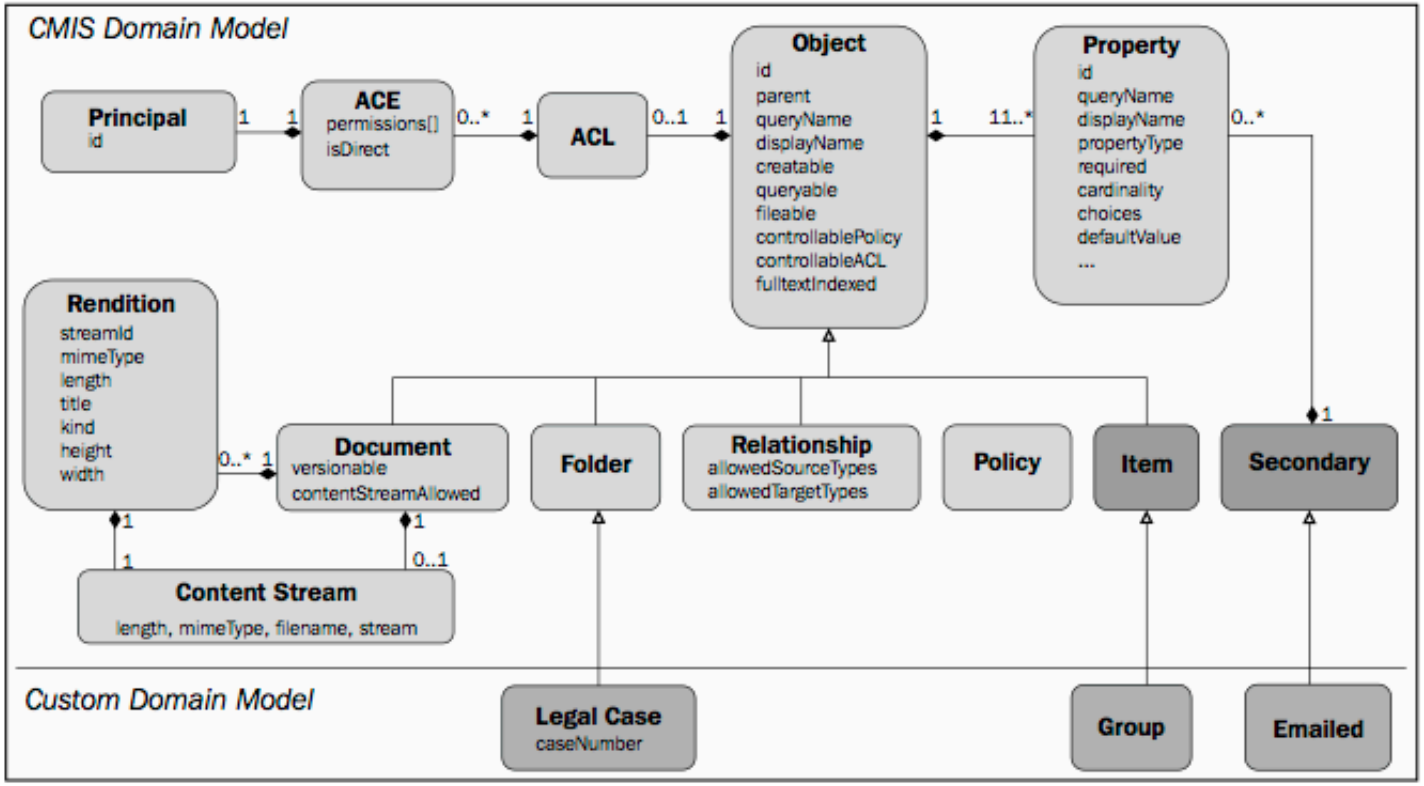
CMIS

Por qué usar CMIS

- Neutralidad tecnológica: cualquier lenguage con soporte para peticiones HTTP y gestión de XML o JSON puede ser utilizado para acceder a un repositorio CMIS
- Independiente de plataforma: no establece ningún requisito para la implementación del repositorio documental. Alfresco está implementado en Java pero Documentum está implementado en C
- API estándar para todas las plataformas
- Lenguaje de búsqueda basado en sintaxis SQL
- Integración directa con sistemas de BPM

CMIS

Modelo de dominio



CMIS

Modelo de dominio

Document

- Objeto de contenido en el repositorio, por un ejemplo un archivo PDF

Folder

- Objeto que contiene otras carpetas o documentos, para cada objeto contenido en la carpeta se crea una relación padre hijo automáticamente

Relationship

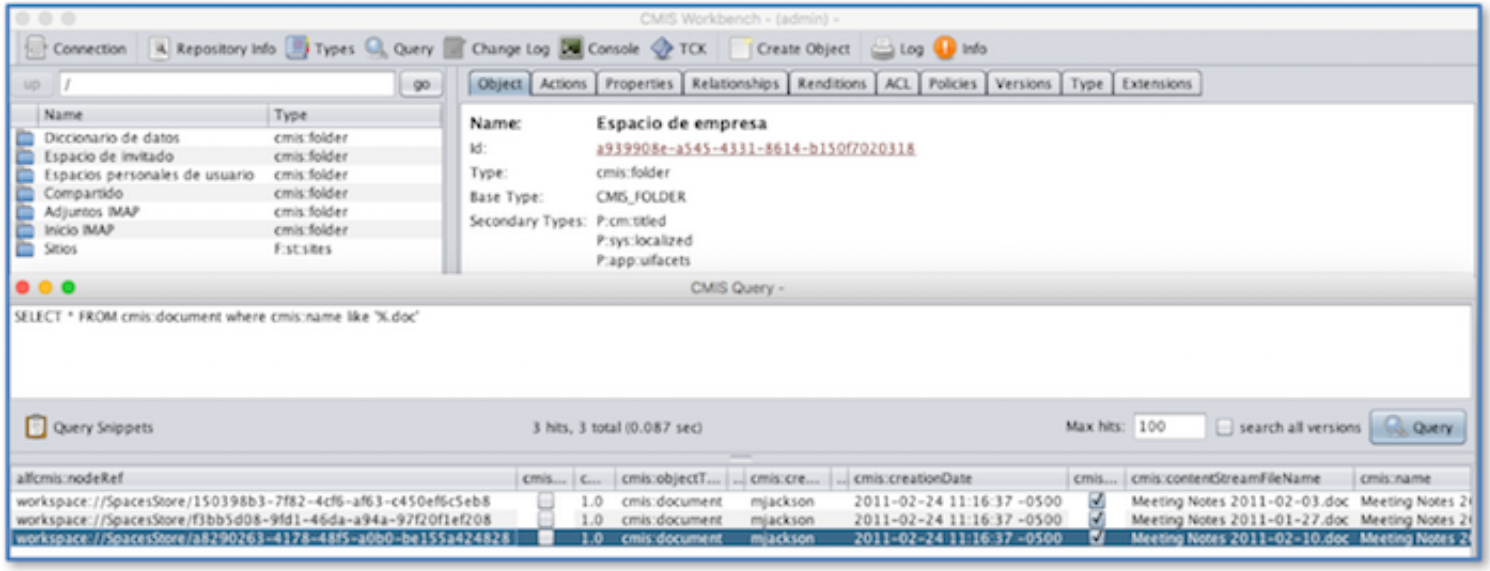
- Representa la relación entre dos objetos origen y destino. Opcional: no implementado en Alfresco

Policy

- Políticas de gestión de objetos, por ejemplo una política de retención

Item

CMIS Workbench



http://chemistry.apache.org/java/developing/tools/dev-tools-workbench.html

CMIS

Servicios

- Repositorio: Información de repositorio y diccionario
- Navegación: Obtener hijos o jerarquías de hijos
- Objetos: Operaciones CRUD
- Descubrimiento: Búsqueda de objetos *queriables*
- Versionado: bloqueo de objetos y recuperación de versiones
- Relaciones: relaciones de objetos
- Políticas: definición y aplicación de políticas
- ACL: permisos de objetos

CMIS

Lenguaje SQL

```
-- All fields from documents
SELECT * FROM cmis:document

-- Some field from documents
SELECT cmis:name, cmis:description FROM cmis:document

-- Get documents having name 'like'
SELECT cmis:name FROM cmis:document WHERE cmis:name LIKE '%contract%'

-- Get documents by using FTS filter
SELECT * FROM cmis:document WHERE CONTAINS('alfresco')

-- Get documents in folder
SELECT * FROM cmis:document WHERE IN_FOLDER('folder id')

-- Get documents in hierarchy
SELECT * FROM cmis:document WHERE IN_TREE('folder id')

-- Get properties from different items
SELECT * FROM cmis:document d JOIN cm:titled t
          ON d.cmis:objectId = t.cmis:objectId
```

CMIS

Protocolos de acceso

Web Services SOAP 1.0

Las peticiones se realizan mediante protocolo SOAP

https://alfresco.keensoft.es/alfresco/cmisws/cmis?wsdl

RESTful AtomPub binding 1.1

Las peticiones se realizan mediante Atom XML feed o Atom XML Entry

https://alfresco.keensoft.es/alfresco/api/-default-/public/cmis/versions/1.1/atom

RESTful Browser binding 1.1

Las peticiones se realizan mediante JSON (en vez de Atom XML feed o Atom XML Entry)

https://alfresco.keensoft.es/alfresco/api/-default-/public/cmis/versions/1.1/browser

CMIS

curl

```
# ATOM - Get repository information
$ curl -u admin:keensoft https://alfresco.keensoft.es/alfresco/cmisatom

# ATOM - Get repository ID
$ curl -u admin:keensoft https://alfresco.keensoft.es/alfresco/cmisatom \
| grep -o "<cmis:repositoryId.*repositoryId"

<cmis:repositoryId>10b21ede-f9b8-4f59-a34f-9a46f854c95a</cmis:repositoryId

# BROWSER - Get repository information
$ curl -u admin:keensoft https://alfresco.keensoft.es/alfresco/cmisbrowser

# BROWSER - Get repository ID
$ curl -u admin:keensoft https://alfresco.keensoft.es/alfresco/cmisbrowser\
|jq '.[] | .repositoryId'

"10b21ede-f9b8-4f59-a34f-9a46f854c95a"
```

CMIS

<) Instala un Alfresco para desarrollo

https://community.alfresco.com/docs/DOC-6296-community-file-list-201605-ga

Windows

`alfresco-community-installer-201605-win-x64.exe`

Mac

`alfresco-community-installer-201605-osx-x64.dmg`

Linux

`alfresco-community-installer-201605-linux-x64.bin`

CMIS

Cliente CMIS Java

Crear un proyecto Maven

```
$ mvn archetype:generate -DgroupId=es.keensoft.cmis -DartifactId=cmis-app \
-DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

Añadir la dependencia del cliente CMIS en `pom.xml`

```
<dependency>
  <groupId>org.apache.chemistry.opencmis</groupId>
  <artifactId>chemistry-opencmis-client-impl
  </artifactId>
  <version>0.14.0</version>
</dependency>
```

CMIS

Cliente CMIS Java

Crear una sesión y recuperar la información del *repositorio*

```
public class App {
  public static void main( String[] args ) {

    SessionFactory sessionFactory = SessionFactoryImpl.newInstance();
    Map<String, String> parameters = new HashMap<String, String>();
    parameters.put(SessionParameter.USER, "admin");
    parameters.put(SessionParameter.PASSWORD, "admin");
    parameters.put(SessionParameter.ATOMPUB_URL,
        "http://localhost:8080/alfresco/api/" +
            "-default-/public/cmis/versions/1.1/atom");
    parameters.put(SessionParameter.BINDING_TYPE, BindingType.ATOMPUB.value());
    parameters.put(SessionParameter.COMPRESSION, "true");
    parameters.put(SessionParameter.CACHE_TTL_OBJECTS, "0");

    Repository repository = sessionFactory.getRepositories(parameters).get(0);
    Session session = repository.createSession();

    System.out.println(session.getRepositoryInfo());

  }
}
```

CMIS

Utilizar el servicio de *navegación*

```
Folder rootFolder = session.getRootFolder();
    for (CmisObject object : rootFolder.getChildren()) {

        if (object.getBaseTypeId() == BaseTypeId.CMIS_FOLDER) {
            Folder folder = (Folder) object;
            System.out.println("Folder: " + folder.getPath());

        } else if (object.getBaseTypeId() == BaseTypeId.CMIS_DOCUMENT) {
            Document document = (Document) object;
            System.out.println("Document: " + document.getName());

        } else {
            System.out.println(object.getBaseType().getDisplayName() + ": " +
                object.getName());
        }
    }
}
```

CMIS

Utilizar el servicio de *objetos*

```
Folder parent = (Folder) session.getObjectByPath("/Sitios/swsdp/documentLibrary");

Map<String, Object> properties = new HashMap<String, Object>();
properties.put(PropertyIds.OBJECT_TYPE_ID, "cmis:document");
properties.put(PropertyIds.NAME, "myNewDocument.txt");

properties.put("cm:description", "My description");
properties.put("cm:title", "My title");

byte[] content = "Hello World!".getBytes();
InputStream stream = new ByteArrayInputStream(content);
ContentStream contentStream =
    new ContentStreamImpl(name, BigInteger.valueOf(content.length),
        "text/plain", stream);

Document doc =
    parent.createDocument(properties, contentStream, VersioningState.MAJOR);
```

CMIS

Utilizar el servicio de *objetos*

```
Map<String, Object> properties = new HashMap<String, Object>();
properties.put(PropertyIds.OBJECT_TYPE_ID, "cmis:document");
properties.put(PropertyIds.NAME, "myNewDocument.txt");

List<Object> aspects = new ArrayList<Object>();
aspects.add("P:cm:titled");
properties.put(PropertyIds.SECONDARY_OBJECT_TYPE_IDS, aspects);

properties.put("cm:description", "My description");
properties.put("cm:title", "My title");

byte[] content = "Hello World!".getBytes();
InputStream stream = new ByteArrayInputStream(content);
ContentStream contentStream =
    new ContentStreamImpl(name, BigInteger.valueOf(content.length),
        "text/plain", stream);

Document doc =
    parent.createDocument(properties, contentStream, VersioningState.MAJOR);
```

CMIS

Utilizar el servicio de *descubrimiento*

```
// false = only last version
ItemIterable<QueryResult> results =
   session.query("SELECT * FROM cmis:document WHERE cmis:name LIKE '%New%'",
       false);

for (QueryResult result : results) {

    String objectId =
        result.getPropertyById("cmis:objectId").getFirstValue().toString();

    Document docResult =
        (Document) session.getObject(session.createObjectId(objectId));

    System.out.println(docResult.getName());

}
```

CMIS

Utilizar el servicio de *versionado*

```
Document docResult =
    (Document) session.getObjectByPath(
    "/Sitios/swsdp/documentLibrary/myNewDocument.txt");

ObjectId pwcId = docResult.checkOut();
Document pwc = (Document) session.getObject(pwcId);

Map<String, Object> properties = new HashMap<String, Object>();
properties.put("cm:title", "My title (version)");

byte[] content = "Hello World (versioned)!".getBytes();
InputStream stream = new ByteArrayInputStream(content);
ContentStream contentStream =
    new ContentStreamImpl(name, BigInteger.valueOf(content.length),
        "text/plain", stream);

// true = Major version
pwc.checkIn(true, properties, contentStream, "Versioned!");

versions = docResult.getAllVersions();
for (Document version : versions) {
    System.out.println("Version: " + version.getVersionLabel());
}
```

CMIS

Otros clientes CMIS

PHP

https://chemistry.apache.org/php/phpclient.html
https://github.com/keensoft/Basic-CMIS

Python

https://chemistry.apache.org/python/cmislib.html

.NET

https://chemistry.apache.org/dotnet/portcmis.html

Objective-C

https://chemistry.apache.org/objective-c/objectivecmis.html

JavaScript

https://chemistry.apache.org/javascript/parts.html (jQuery)

CMIS

<) Revisión: Lista todas las categorías de la rama Regiones

Puedes obtener el nodo raíz mediante un PATH FTS.

```
ItemIterable<QueryResult> regionRoot =
    session.query(
        "SELECT * FROM cm:category C WHERE " +
        "CONTAINS(C, 'PATH:\"/cm:generalclassifiable/cm:Regiones\"')", false);
```

Utiliza el método `IN_TREE` para recuperar todos los nodos contenidos en la categoría `Regiones`

```
ItemIterable<QueryResult> regions =
    session.query("...", false);
```

Imprime la lista de resultados

```
for (QueryResult region : regions) {
    Object nodeRef = region.getPropertyValueById("alfcmis:nodeRef");
    String nameCategory = region.getPropertyValueById("cm:name").toString();
    System.out.println("NodeRef: " + nodeRef + ", name: " + nameCategory);
}
```

REST API

REST API

Alfresco 5.1 o anterior

API nativa desarrollada mediante tecnología Web Script

https://alfresco.keensoft.es/alfresco/s/index/uri/

Solo los que comienzan por **/api** son considerados de la API pública

Alfresco 5.2 o posterior

Basada en estándar OpenAPIs y definida en YAML

https://alfresco.keensoft.es/api-explorer-1.2/

https://github.com/Alfresco/rest-api-explorer/tree/master/src/main/webapp/definitions

REST API

API nativa

Sites

GET https://alfresco.keensoft.es/alfresco/s/api/sites

GET https://alfresco.keensoft.es/alfresco/s/api/sites/addons

GET | POST | PUT
https://alfresco.keensoft.es/alfresco/s/api/sites/addons/memberships

GET | POST | PUT | DELETE
https://alfresco.keensoft.es/alfresco/s/api/sites/addons/memberships/admin

GET https://alfresco.keensoft.es/alfresco/s/api/sites/addons/roles

GET | POST https://alfresco.keensoft.es/alfresco/s/api/sites/addons/invitations

GET https://alfresco.keensoft.es/alfresco/s/api/sites/addons/export

REST API

API nativa

People

GET | POST https://alfresco.keensoft.es/alfresco/s/api/people

GET https://alfresco.keensoft.es/alfresco/s/api/people/admin

GET https://alfresco.keensoft.es/alfresco/s/api/people/admin/sites

GET | POST | DELETE
https://alfresco.keensoft.es/alfresco/s/api/people/admin/preferences

POST https://alfresco.keensoft.es/alfresco/s/api/person/changepassword/admin

REST API

API nativa

Group

GET https://alfresco.keensoft.es/alfresco/s/api/groups

GET | PUT | DELETE
https://alfresco.keensoft.es/alfresco/s/api/groups/ALFRESCO_ADMINISTRATORS

GET
https://alfresco.keensoft.es/alfresco/s/api/groups/ALFRESCO_ADMINISTRATORS/children

GET
https://alfresco.keensoft.es/alfresco/s/api/groups/ALFRESCO_ADMINISTRATORS/children/a

GET
https://alfresco.keensoft.es/alfresco/s/api/groups/ALFRESCO_ADMINISTRATORS/parents

REST API

API nativa

Tags

GET | POST
https://alfresco.keensoft.es/alfresco/s/api/tags/workspace/SpacesStore

DELETE
https://alfresco.keensoft.es/alfresco/s/api/tags/workspace/SpacesStore/ejemplos

GET
https://alfresco.keensoft.es/alfresco/s/api/tags/workspace/SpacesStore/ejemplos/nodes

REST API

API nativa **Nodes**

GET
https://alfresco.keensoft.es/alfresco/s/api/node/workspace/SpacesStore/7e49b7c5-067b-4cfb-8eec-ba46d7908ba7/metadata

POST
https://alfresco.keensoft.es/alfresco/s/api/metadata/node/workspace/SpacesStore/7e49b7c
067b-4cfb-8eec-ba46d7908ba7

GET | POST
https://alfresco.keensoft.es/alfresco/s/api/node/workspace/SpacesStore/7e49b7c5-067b-4cfb-8eec-ba46d7908ba7/comments

GET | POST
https://alfresco.keensoft.es/alfresco/s/api/node/workspace/SpacesStore/7e49b7c5-067b-4cfb-8eec-ba46d7908ba7/tags

GET
https://alfresco.keensoft.es/alfresco/s/api/node/workspace/SpacesStore/7e49b7c5-067b-4cfb-8eec-ba46d7908ba7/ruleset/rules

REST API

API nativa

Content

GET
https://alfresco.keensoft.es/alfresco/s/api/node/content/workspace/SpacesStore/755647d1bc54-4c2f-9235-61e200653b24

POST https://alfresco.keensoft.es/alfresco/s/api/upload

```
$ curl -u admin:admin -v POST -F filedata=@test.txt -F siteid=swsdp \
-F containerid=documentLibrary -F uploaddirectory=/ \
http://localhost:8080/alfresco/service/api/upload

{
   "nodeRef": "workspace://SpacesStore/5dab255b-1187-4ea0-8e9a-19fa6772ef40",
   "fileName": "test.txt",
   "status":
   {
      "code": 200,
      "name": "OK",
      "description": "File uploaded successfully"
   }
}
```

REST API

API nativa

Tasks

GET https://alfresco.keensoft.es/alfresco/s/api/workflow-definitions

GET https://alfresco.keensoft.es/alfresco/s/api/workflow-instances

GET https://alfresco.keensoft.es/alfresco/s/api/node/workspace/SpacesStore/7e49b7c5-067b-4cfb-8eec-ba46d7908ba7/workflow-instances

POST https://alfresco.keensoft.es/alfresco/s/api/workflow/task/end/4185

REST API

API nativa

Cliente Java

Añadir la dependencia del cliente HTTP en el fichero **pom.xml** empleado en el ejemplo anterior

```
<dependency>
    <groupId>org.apache.httpcomponents</groupId>
    <artifactId>httpcore</artifactId>
    <version>4.1</version>
</dependency>
<dependency>
    <groupId>org.apache.httpcomponents</groupId>
    <artifactId>httpmime</artifactId>
    <version>4.1</version>
</dependency>
<dependency>
    <groupId>org.apache.httpcomponents</groupId>
    <artifactId>httpclient</artifactId>
    <version>4.1</version>
</dependency>
```

REST API

API nativa

Cliente Java

Crear un método `main` que recupere un ticket de sesión de Alfresco (1)

```
// HTTP request

JSONObject au = new JSONObject();
au.put("username", "admin");
au.put("password", "admin");

HttpClient httpClient = new DefaultHttpClient();
HttpPost postRequest =
        new HttpPost("http://localhost:8080/alfresco/service/api/login");

StringEntity input = new StringEntity(au.toJSONString());
input.setContentType("application/json");
postRequest.setEntity(input);

HttpResponse response = httpClient.execute(postRequest);
```

Sigue >>

REST API

API nativa

Cliente Java

Crear un método `main` que recupere un ticket de sesión de Alfresco (2)

```
// HTTP response

JSONParser jsonParser = new JSONParser();
JSONObject ticket =
        (JSONObject) jsonParser.parse(
                IOUtils.readAllLines(response.getEntity().getContent()
                )
            );
JSONObject data = (JSONObject) ticket.get("data");
String alfrescoTicket = data.get("ticket").toString();
System.out.println(alfrescoTicket);

httpClient.getConnectionManager().shutdown();
```

REST API

API nativa

Cliente Java

Invocar al servicio de lista de Sitios

```
httpClient = new DefaultHttpClient();
httpClient.getParams().setParameter("http.protocol.version",
    HttpVersion.HTTP_1_1);
httpClient.getParams().setParameter("http.protocol.content-charset", "UTF-8");

HttpGet getRequest =
        new HttpGet("http://localhost:8080/alfresco/s/api/sites?alf_ticket=" +
            alfrescoTicket);

response = httpClient.execute(getRequest);
JSONParser jsonParser = new JSONParser();
JSONArray body =
    (JSONArray) jsonParser.parse(IOUtils.readAllLines(
        response.getEntity().getContent()));

for (Object element : body) {
    JSONObject site = (JSONObject) element;
    System.out.println(site.get("shortName"));
}
```

REST API

<) Revisión: Obtén la lista todos los `userName` del sistema con el cliente Java

http://localhost:8080/alfresco/s/api/people


```
{
  "people" : [
    {
        "url": "\/alfresco\/s\/api\/people\/guest",
        "userName": "guest",
        "enabled": false,
        "firstName": "Guest",
        "lastName": ""
    }
    ,
    {
        "url": "\/alfresco\/s\/api\/people\/abeecher",
        "userName": "abeecher",
        "enabled": false,
        "firstName": "Alice",
        "lastName": "Beecher"
    }
  ]
}
```

REST API

OpenAPIs

Explorar y probar la API

https://alfresco.keensoft.es/api-explorer-1.2

Explorar la definición de los servicios en YAML

https://github.com/Alfresco/rest-api-explorer/tree/master/src/main/webapp/definitions

Crear un cliente Java para el core

https://raw.githubusercontent.com/Alfresco/rest-api-explorer/master/src/main/webapp/definitions/alfresco-core.yaml

http://editor.swagger.io/

```
Generate Client > Java
```

REST API

OpenAPIs

Obtener una lista de sitios

```
ApiClient apiClient = Configuration.getDefaultApiClient();
apiClient.setBasePath(
"https://alfresco.keensoft.es/alfresco/api/-default-/public/alfresco/versions/1");
apiClient.setUsername("admin");
apiClient.setPassword("keensoft");

SitesApi api = new SitesApi();
api.setApiClient(apiClient);
SitePaging sites = api.listSites(null, null, null, null, null, null);
for (SiteEntry site : sites.getList().getEntries()) {
    System.out.println(site.getEntry().getId());
}
```

REST API

<) Revisión: Obtén la lista todos los `userName` del sistema con el cliente Swagger

Utiliza el *handler* generado por Swagger `PeopleApi`

La URL REST que invoca es la siguiente:

https://alfresco.keensoft.es/alfresco/api/-default-/public/alfresco/versions/1/people

Aikau

Aikau

Aikau es una tecnología de desarrollo web creado por Alfresco que se utiliza sobre todo en los productos Alfresco Share y Alfresco Record Management.

Este tecnología permite crear aplicaciones web sobre Alfresco de manera rápida mediante un conjunto de componentes pre-configurados.

Puede consultarse el detalle de estos componentes en
http://dev.alfresco.com/resource/docs/aikau-jsdoc/

Aikau

Puede crearse una aplicación web Aikau a través de un arquetipo Maven

```
$ mvn archetype:generate -DgroupId=es.keensoft.cmis -DartifactId=aikau-app \
-DarchetypeCatalog=https://artifacts.alfresco.com/nexus/content/groups/public/arch
-DarchetypeGroupId=org.alfresco -DarchetypeArtifactId=aikau-sample-archetype \
-DarchetypeVersion=RELEASE -DinteractiveMode=false
```

Una vez configurada, se compila y se lanza desde la carpeta `aikau-app`

```
$ mvn install
$ mvn jetty:run
```

Por defecto requiere que un repositorio de Alfresco esté disponible en el puerto 8080

http://localhost:8080/alfresco

Una vez arrancada, es accesible a través de la siguiente URL

http://localhost:8090/aikau-sample/

Aikau

`/src/main/webapp/WEB-INF/webscripts/pages/home.get.js`

Las páginas se construyen declarando *widgets* basados en los componentes de construcción de Aikau...

```
// Add more widgets here !!!
,{
  name: "alfresco/documentlibrary/AlfDocumentList",
  config: {
    rootNode: "alfresco://user/home",
    rawData: true,
    widgets: [
      {
        name: "alfresco/documentlibrary/views/AlfSimpleView"
      }
    ]
  }
}
```

... y asociándolos a servicios de datos

```
// Add more services here !!!
,"alfresco/services/DocumentService"
```

Aikau

Pueden encontrarse ejemplos de utilización detallados de los componentes en Aikau Sandpit

https://aikau-sandpit.alfresco.com/aikau-sandpit/page/na/ws/home

Existen tutoriales disponibles

https://github.com/Alfresco/Aikau/blob/master/tutorial/chapters

Y una buena colección de artículos

https://community.alfresco.com/community/ecm/blog/tags#/?tags=aikau

La tecnología YUI de Alfresco Share está siendo reemplaza paulatinamente por páginas Aikau

https://community.alfresco.com/community/ecm/blog/2016/11/23/create-and-edit-site-customization

Aikau

Mediante mecanismos de autenticación como Kerberos SSO, las aplicaciones web Aikau pueden integrarse de manera transparente en las organizaciones

https://angelborroy.wordpress.com/2016/10/05/sso-support-for-aikau-apps/

https://github.com/angelborroy/aikau-kerberos-sso

ADF

ADF

Application Development Framework es la nueva plataforma de desarrollo de Alfresco basada en *Angular 2* y *Google Material*

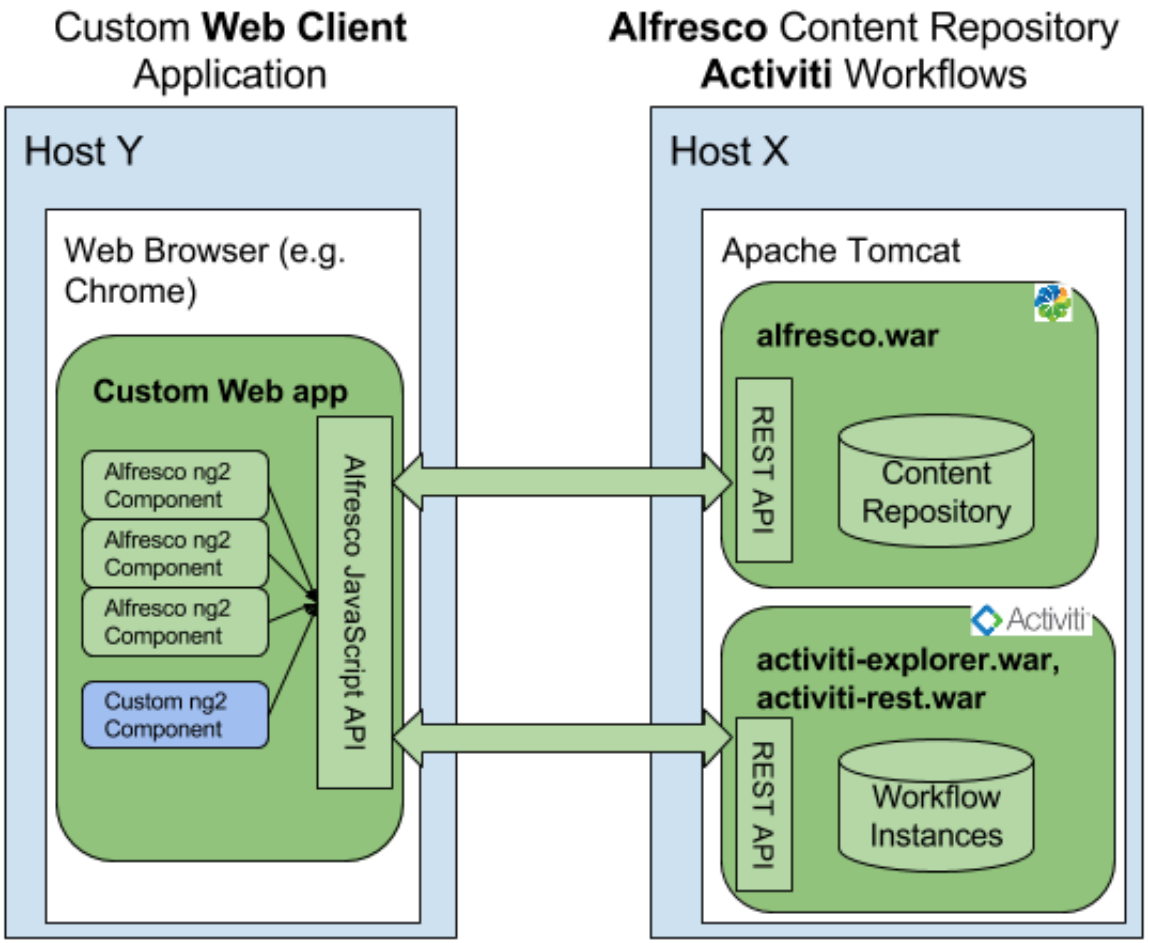
Estará disponible a partir de Alfresco 5.2, ya que funciona sobre la nueva API REST basada en *OpenAPIs*

Permite construir aplicaciones Angular 2 a partir de componentes web prefabricados:

- Core library
- DataTable
- DocumentList
- Viewer
- Login
- Upload

Referencia de los componentes: http://devproducts.alfresco.com/

ADF


Custom **Web Client** Application
Alfresco Content Repository **Activiti** Workflows
Host Y
Host X
Web Browser (e.g. Chrome)
Apache Tomcat
Custom Web app
Alfresco ng2 Component
Alfresco ng2 Component
Alfresco ng2 Component
Custom ng2 Component
Alfresco JavaScript API
alfresco.war
REST API
Content Repository
Activiti
activiti-explorer.war, activiti-rest.war
REST API
Workflow Instances

ADF

De manera habitual las aplicaciones desarrolladas con ADF se instalan en un servidor distinto al que ejecuta Alfresco, ya que se trata de tecnologías diferentes

Por este motivo, es necesario habilitar CORS en el servidor de Alfresco

```
$ wget https://artifacts.alfresco.com/nexus/service/local/repositories/releases/
content/org/alfresco/enablecors/1.0/enablecors-1.0.jar
$ cp enablecors-1.0.jar /opt/alfresco/modules/platform
$ service alfresco restart
```

Requiere una versión de `node` superior a la 5.12

```
$ node -v
v6.2.2
```

Para la construcción de aplicaciones utiliza el generador Yeoman, que también debe ser instalado en la máquina de desarrollo

```
$ npm install -g yo
```

ADF

Alfresco provee un generador de aplicaciones basado en Yeoman

```
$ npm install -g generator-ng2-alfresco-app
```

Una vez instalado, puede construirse la aplicación de prueba.

```
$ yo ng2-alfresco-app
? What's the name of your App? ng2-test
Your generator must be inside a folder named ng2-test
I'll automatically create this folder.
? How would you describe the app? Alfresco Angular 2 Application Example
? Author's Name Angel Borroy
? Author's Email angel.borroy@keensoft.es
? Author's Homepage
? Package keywords (comma to split)
? What is your Alfresco platform server URL? http://localhost:8080
? What is your Activiti platform server URL? http://127.0.0.1:9999
? GitHub username or organization
? Do you want include the User info component? Yes
? Do you want include a drawer bar? Yes
? Do you want include a search bar? Yes
? Do you want include a Document List? Yes
? Do you want include a Tasks List? Yes
? Which license do you want to use? Apache 2.0
```

ADF

Una vez generada, la aplicación puede lanzarse mediante *Node.js*

```
$ cd ng2-test
$ npm update
$ npm start
```

http://localhost:3000

Más información

https://community.alfresco.com/docs/DOC-4595-getting-started-with-alfresco-application-development-framework

Formación Alfresco

keensoft

Día 2 - Integración