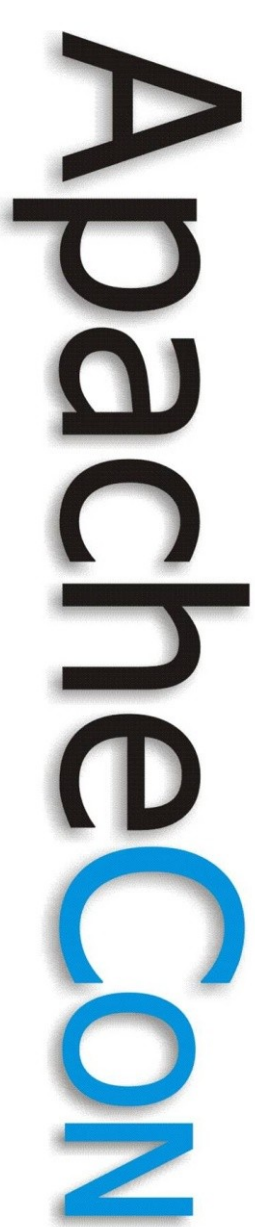
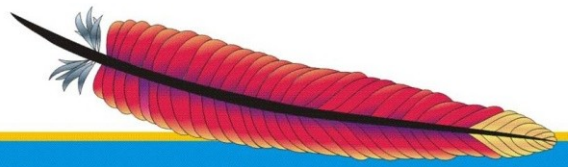
PhotArk/Tuscany Integration

Avdhesh Yadav
avd@apache.org

Luciano Iresende
lresende@apache.org





Who Am I

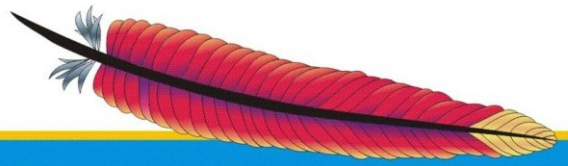
- Committer Since Feb 2010.

Agenda

- Tuscany Overview
- PhotArk Introduction
- Integration with Tuscany
- PhotArk Components
- Demo
- Getting Involved

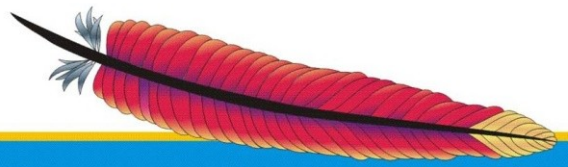
Tuscany Overview

- SCA is set of specifications defined by OASIS.
- Tuscany implements the SCA specifications.
- Tuscany Simplyfies the building of SOA applications and implements SCA standards.
- Decouples the infrastructure from the business logic.
- Declartive approach speeds up the development.
- Simple XML constructs used to assemble the Services together.
- Assembly Model is the base of the Apache Tuscany.



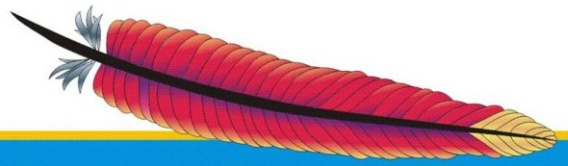
Tuscany Overview

- Reusable business functions can be easily developed and tested.
- The Resuble components can be easily assembled together to build application.
- Tuscany provides higher level of abstraction to the services.
- Components can be implemented in different programming languages and connected togther using Apache Tuscany.



Challenges

- Integration with web 2.0 technolgies.
- Reusablty.
- Business logic mixed with infrastruture.
- Multiple deployment scenarios.
    - Tomcat.
    - Google Apps Engine.

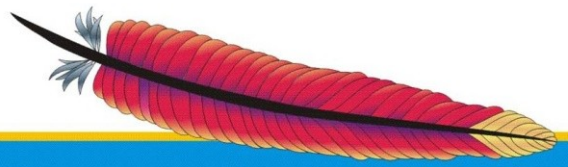


PhotArk Introduction

- Apache PhotArk is a open source photo gallery application including a content repository for the images, a display piece, an access control layer, and upload capabilities
- PhotArk application is designed as a Web 2.0 application that can deployed to a Servlet Engine such as Apache Tomcat or Cloud Environment such as Google App Engine.
- Albums can be hosted locally but we also support subscription to remote albums(e.g Picasa, Flickr etc).

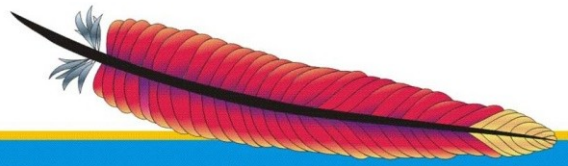
PhotArk Introduction

- The PhotArk components are defined using the SCA programming model and Apache Tuscany is used as the SCA Runtime
- Apache JackRabbit used for the storing of contents.

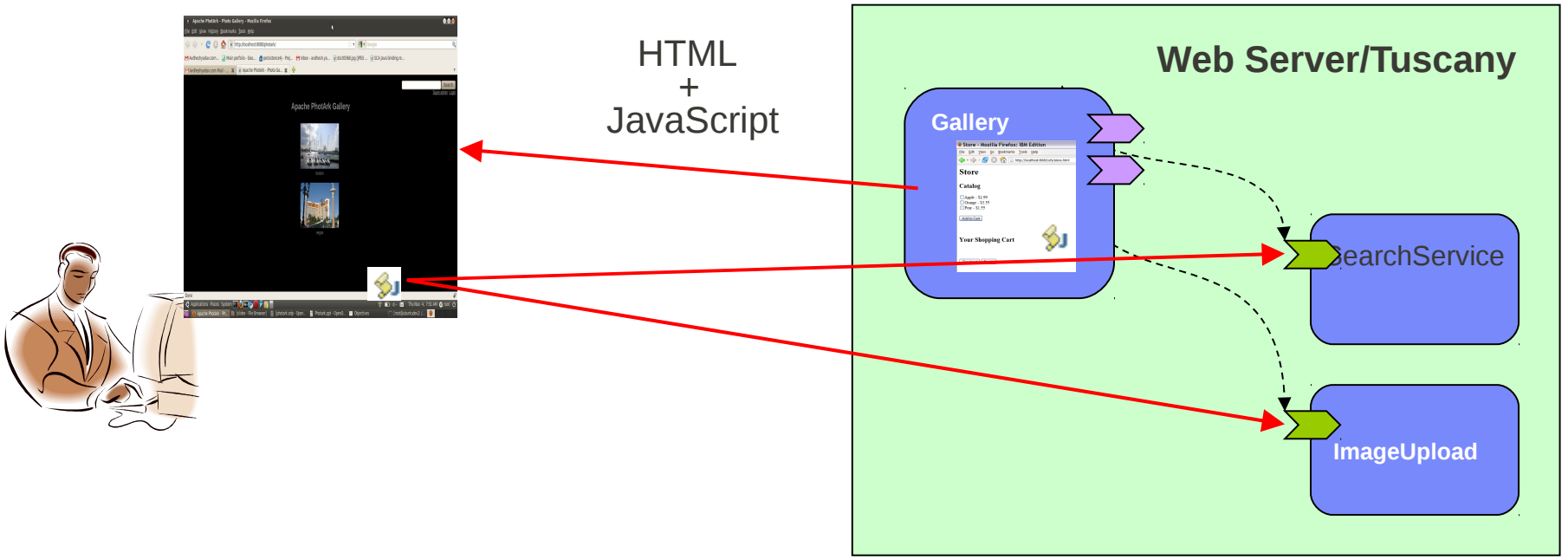


PhotArk Introduction

- PhotArk has Access layer for protecting the albums.
- Open Id support for user authentication.
- Lucene integration for searching and tagging.

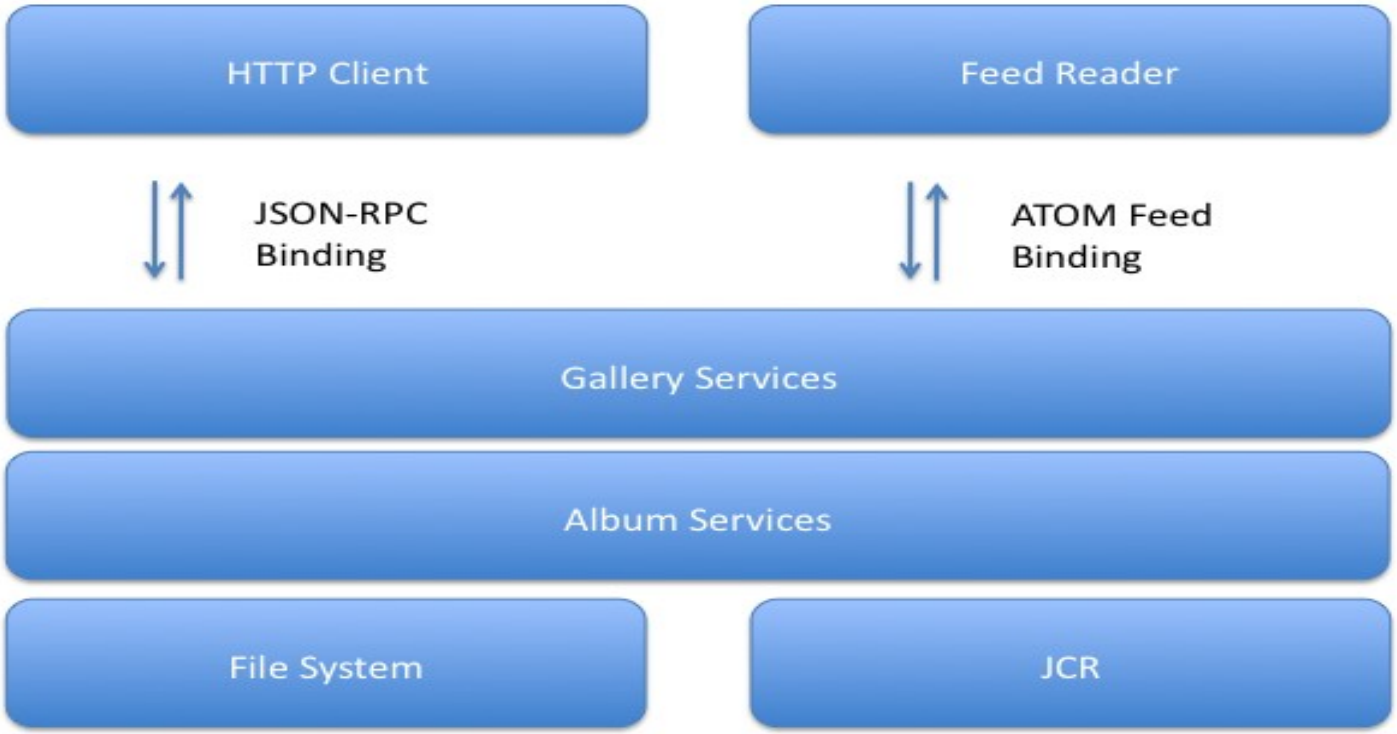


Web 2.0 & Tuscany



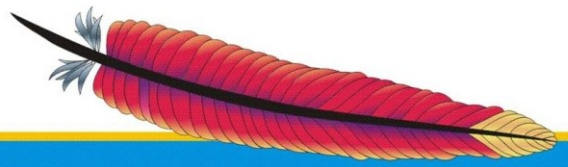
- Simplifies consumption of SCA Business Services
  - SCA References, Dependency Injection, etc
  - JavaScript Framework like Dojo can easily consume Services.
- Developer concentrate in business logic
- Protocols becomes a choice

Architecture



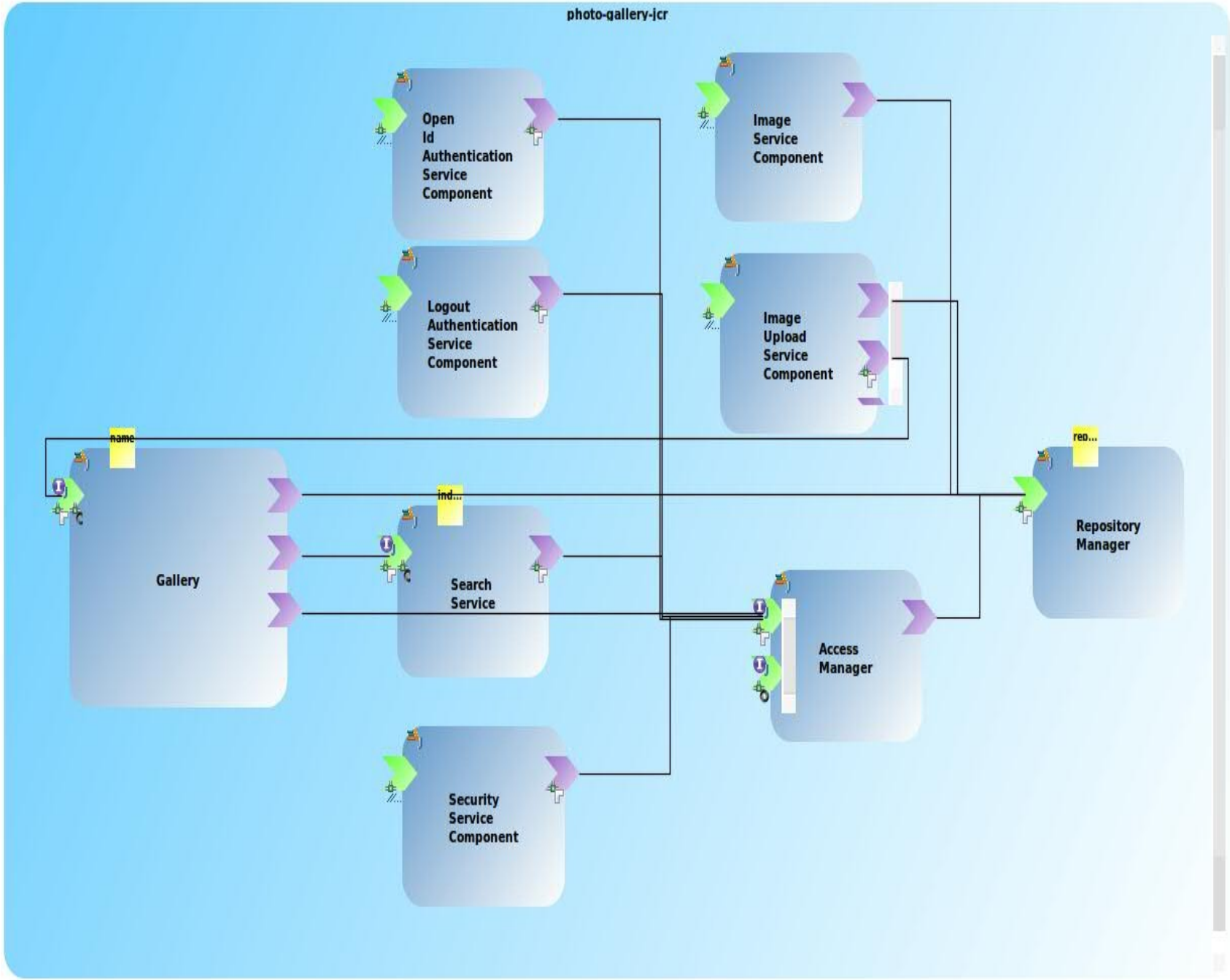
Tuscany Integration

- Tuscany Helped in creating PhotArk functions as resuable SCA components
- Dependecies on other components injected through references.
- Bindings helped in decoupling the protocols from the buisness functions.
- PhotArk components exposes services using bindings(e.g JSON-RPC,REST).



PhotArk Composite

```
<!-- Facade component for Gallery -->
   <component name="Gallery">
          <implementation.java class="org.apache.photark.jcr.services.JCRGaller
          <property name="name">gallery</property>
          <service name="Gallery">
               <interface.java interface="org.apache.photark.services.gallery.Gal
               <binding.sca name="local"/>
               <tuscany:binding.jsonrpc uri="/GalleryService"/>
          </service>
          <reference name="repositoryManager" target="RepositoryManager"/>
          <reference name="listeners" target="SearchService"/>
        <reference name="accessmanager" target="AccessManager"/>
   </component>
   <!-- Component responsible for providing REST access to images -->
   <component name="ImageServiceComponent">
          <implementation.java class="org.apache.photark.jcr.services.JCRImage
          <service name="ImageCollection">
               <tuscany:binding.http uri="/gallery"/>
          </service>
          <reference name="repositoryManager" target="RepositoryManager"/>
     </component>
   <!-- Component responsible for providing upload support for gallery/album -->
   <component name="ImageUploadServiceComponent">
          <implementation.java class="org.apache.photark.jcr.services.JCRImage
          <service name="Servlet">
               <tuscany:binding.http uri="/admin/upload"/>
          </service>
          <reference name="repositoryManager" target="RepositoryManager"/>
          <reference name="gallery" target="Gallery">
               <binding.sca name="local"/>
          </reference>
          <reference name="accessmanager" target="AccessManager"/>
     </component>
   <!-- Component responsible for providing JCR Management Support -->
   <component name="RepositoryManager">
          <implementation.java class="org.apache.photark.jcr.JCRRepositoryMan
          <property name="repositoryHome">photark</property>
   </component>
```



ApacheCon

Leading the Wave of Open Source

Bindings

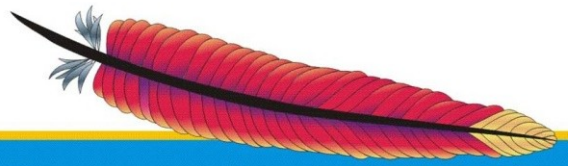
- Tuscany support JSON-RPC binding which enables remote web applications to easily make RPC calls to the server-side PhotArk services.

```
<tuscany:binding.jsonrpc uri="/GalleryService"/>
```

- PhotArk exploits Dojo native support for JSON-RPC services.

```
var  gallery = new dojo.rpc.JsonService( /GalleryService?smd );
```

```
<component name="Gallery">
 <implementation.java class="org.apache.photark.jcr.services.JCRGalleryImpl"/>
 <property name="name">gallery</property>
 <service name="Gallery">
  <interface.java interface="org.apache.photark.services.gallery.Gallery"/>
  <binding.sca name="local"/>
   <tuscany:binding.jsonrpc uri="/GalleryService"/>
 </service>
 <reference name="repositoryManager" target="RepositoryManager"/>
 <reference name="listeners" target="SearchService"/>
 <reference name="accessmanager" target="AccessManager"/>
</component>
```



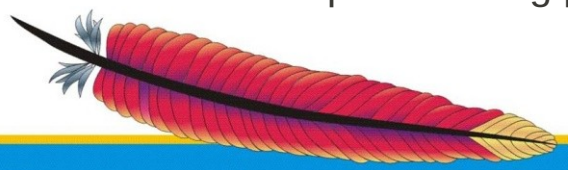
Bindings

- Tuscany out of box support REST.PhotArk implements Collection Interface that matches the actions of HTTP protocol.

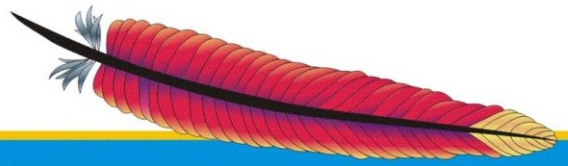
```
<!-- Component responsible for providing REST access to images -->
<component name="ImageServiceComponent">
  <implementation.java class="org.apache.photark.jcr.services.JCRImageCollectionImpl"/>
  <service name="ImageCollection">
      <tuscany:binding.http uri="/gallery"/>
</service>
<reference name="repositoryManager" target="RepositoryManager"/>
</component>
```

Collection Interface Implementation

```
public InputStream get(String key) throws NotFoundException {
        String sub = StringUtils.substringAfter(key, "gallery/");
        String stringArray[] = StringUtils.split(sub, "/");
        String albumName = stringArray[0];
        InputStream inStream = null;
        Session session = repositoryManager.getSession();
         Node root = session.getRootNode();
         Node albumNode = root.getNode(albumName);
         String image = stringArray[1];
         Node picNode = albumNode.getNode(image);
         inStream = picNode.getProperty("imageContent").getStream();
        return inStream;
    }
    public void delete(String key) {};
    public void put(String key, InputStream inputStream) {}
    public String post(String key, InputStream inputStream) {return key }
```

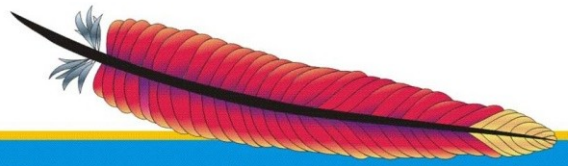


Demo



What's Next

- Work on REST API.
- Add Subscription UI.
- Enhance the feature set when deployed on Google Apps Engine.



Getting Involved

- Apache PhotArk
  - http://incubator.apache.org/photark
- Apache PhotArk Getting started

  http://incubator.apache.org/photark/photark-developer-guide.htm
- Apache Photark Documentaton
  -

  http://incubator.apache.org/photark/photark-documentation.html
- Apache Tuscany
  - http://tuscany.apache.org

Thank You !