BACK-PORTING DSPACE 2.0: DSPACE SERVICES

Mark Diggory  DSUG 2009

Thursday, November 19, 2009
Conflicts in DSpace. Developers vs. Developers?
But is it Their Fault?

Pioneer Argonne computer scientist Jean F. Hall.
But is it Their Fault?

No, Developers:

Pioneer Argonne computer scientist Jean F. Hall.
But is it Their Fault?

No, Developers:

Need to Innovate

Pioneer Argonne computer scientist Jean F. Hall.
But is it Their Fault?

No, Developers:
Need to Innovate
Need to change code

Pioneer Argonne computer scientist Jean F. Hall.
But is it Their Fault?

No, Developers:
Need to Innovate
Need to change code
Need to solve immediate issues foremost.

Pioneer Argonne computer scientist Jean F. Hall.
Static code is not extensible...

```java
public class StaticManager {
    public static Object getSomething(Object object) {
        SomeOtherManager.doSomethingElse(...);
    }
}
```
Consider Anti-Patterns
Consider Anti-Patterns

- **Hardcoding:**
  Configuration is hardcoded into static “Managers”
  Database CRUD is hardcoded into DpaceObjects.”
**Consider Anti-Patterns**

- **Hardcoding:**
  - Configuration is hardcoded into static “Managers”
  - Database CRUD is hardcoded into DspaceObjects.

- **God Object:**
  - ConfigurationManager, Context, DSpaceObject
  - Concentrate too much functionality in a class
Consider Anti-Patterns

- **Hardcoding:**
  Configuration is hardcoded into static “Managers”
  Database CRUD is hardcoded into DspaceObjects."

- **God Object:**
  ConfigurationManager, Context, DSpaceObject
  Concentrate too much functionality in a class

- **JAR Hell:**
  Users resort to classpath ordering to overload core API.
  User override classes directly to change behavior.
Importance to @mire?
Importance to @mire?

- Many clients with similar need for customization.
Importance to @mire?

- Many clients with similar need for customization.
- All Products dependent on DSpace.
Importance to @mire?

- Many clients with similar need for customization.
- All Products dependent on DSpace.
- We have to guarantee upgrade path.
Importance to @mire?

- Many clients with similar need for customization.
- All Products dependent on DSpace.
- We have to guarantee upgrade path.
- Need stability and modularity in DSpace.
Modularity
Modularity

Modularity


Services: Can Help
Services: Can Help

- **Removes Hardcode:**
  Data Models are anemic, Services implemented separate from interfaces used by applications.
Services: Can Help

- **Removes Hardcode:**
  Data Models are anemic, Services implemented separate from interfaces used by applications.

- **Lessens JAR Hell:**
  API contracts, default implementations off limits.
  Want to change behavior, write changes separately.
Services: Can Help

- **Removes Hardcode:**
  Data Models are anemic, Services implemented separate from interfaces used by applications.

- **Lessens JAR Hell:**
  API contracts, default implementations off limits. Want to change behavior, write changes separately.

- **Removes God Objects:**
  Services separate functional areas, separate Data Models without interdependency assure separation.
Services: Architecture:

- services-util
- services-api
- services-impl
Services: Architecture:

- **services-util**
  - DSpaceWebapp-ServletFilter

- **services-api**
  - DSpaceKernelManager
  - DSpaceKernel

- **services-impl**
  - DSpaceKernel-ServletContextListener
  - DSpaceKernelInit
  - DSpaceKernelImpl

**JMX**
Services: Architecture:

- services-util
- services-api
- services-impl

DSpace

<<Interface>>
ServiceManager

<<Interface>>
EventService
/* Instantiate the Utility Class */
DSpace dspace = new DSpace();

/* Access get the Service Manager by convenience method */
ServiceManager manager = dspace.getServiceManager();

/* Or access by convenience method for default services */
EventService service = dspace.getEventService();
Services: Default Services

services-util
- DSpace

services-api
- <<Interface>> ServiceManager
- <<Interface>> EventService
- <<Interface>> ConfigurationService
- <<Interface>> RequestService
- <<Interface>> SessionService
DSpace dspace = new DSpace();

EventService es = dspace.getEventService();
ConfigurationService cs = dspace.getConfigurationService();
RequestService rs = dspace.getRequestService();
SessionService ss = dspace.getSessionService();
Services: Default Services

- Default Services
  - EventService
  - ServicesManager
  - services-api
    - ConfigurationService
    - RequestService
    - SessionService
  - services-util
    - DSpace
- Your Own Services
- Your Own Event Listeners
- Your Own Configs
- Your Own Request Interceptors

Register Service: registerService
Register Event Listener: registerEventListener
Add Configuration: addConfiguration
Add Interceptor: addInterceptor

Spring Application Context

Yours Own Services
Yours Own Configs
Yours Own Event Listeners
Yours Own Request Interceptors

Thursday, November 19, 2009
Spring:
Web Application Context
<beans>
  
  <bean id="dspace" class="org.dspace.utils.DSpace"/>
  
  <bean id="dspace.eventService" factory-bean="dspace"
       factory-method="getEventService"/>
  
  <bean class="org.my.EventListener">
    
    <property name="eventService">
      <ref bean="dspace.eventService"/>
    </property>
  
  </bean>
</beans>

<?xml version="1.0" encoding="UTF-8"?>
<beans>
  
  <bean id="dspace" class="org.dspace.utils.DSpace"/>
  
  ...         
  
  <property name="eventService">
    
    <ref bean="dspace.eventService"/>
  
  </property>
  
</bean>
DSpace dspace = new DSpace();

EventService service = dspace.getEventService();

MyEventListener listener = new MyEventListener();

service.registerEventListener(listener);
Our first example of service usage

DSpace 1.6 Statistics

HTTP Request

Cocoon

UsageEvent
LoggingActor

EventService

SolrUsage
EventListener

SolrLogger

DSpace XMLUI Webapp

HTTP
Request

receiveEvent

post

HTTP

Solr

writes to log

dspace
.log

receiveEvent

receiveEvent
DSpace dspace = new DSpace();

Event event = new UsageEvent(
    UsageEvent.Action.VIEW,
    request,
    context,
    object);

dspace.getEventService().fireEvent(event);
Proposed Next Steps:
Integrate remaining Services
Proposed Next Steps:
Integrate remaining Services

- DSpaceDataSource: DB Connection Pool
Proposed Next Steps:
Integrate remaining Services

- **DSpaceDataSource**: DB Connection Pool
- **UserService**: Auth and Permissions
Proposed Next Steps:
Integrate remaining Services

- **DSpaceDataSource**: DB Connection Pool
- **UserService**: Auth and Permissions
- **StorageService**: ContentStorage
Proposed Next Steps:

Integrate remaining Services

- **DSpaceDataSource**: DB Connection Pool
- **UserService**: Auth and Permissions
- **StorageService**: ContentStorage
- **MetaRegistryService**: Content Models, Metadata Schema, DCMI Application Profiles.
Proposed Next Steps:

Integrate remaining Services

- **DSpaceDataSource**: DB Connection Pool
- **UserService**: Auth and Permissions
- **StorageService**: ContentStorage
- **MetaRegistryService**: Content Models, Metadata Schema, DCMI Application Profiles.
- **SearchService**: Unified search and browse
Proposed Next Steps:

Integrate remaining Services

- **DSpaceDataSource**: DB Connection Pool
- **UserService**: Auth and Permissions
- **StorageService**: ContentStorage
- **MetaRegistryService**: Content Models, Metadata Schema, DCMI Application Profiles.
- **SearchService**: Unified search and browse
- **MappingService**: External Identifier Mapping to DSpace objects.
Proposed Next Steps:
Replacement of Legacy Managers
Proposed Next Steps:

Replacement of Legacy Managers

- EventManager <- EventService
Proposed Next Steps:
Replacement of Legacy Managers

- EventManager <----------------------------- EventService
- ConfigurationManager <---------------------- ConfigurationService
Proposed Next Steps:
Replacement of Legacy Managers

- EventManager <----------------------------- EventService
- ConfigurationManager <---------------------- ConfigurationService
- DatabaseManager <-------------------------- DSpaceDataSource Service
Proposed Next Steps:

Replacement of Legacy Managers

- EventManager <----------------------------- EventService
- ConfigurationManager <--------------------- ConfigurationService
- DatabaseManager <------------------------ DSpaceDataSource Service
- EPerson/ResourceBundle <------------------ UserService

Thursday, November 19, 2009
Proposed Next Steps:

Replacement of Legacy Managers

- EventManager ←-------------------------- EventService
- ConfigurationManager ←--------------- ConfigurationService
- DatabaseManager ←------------------- DSpaceDataSource Service
- EPerson/ResourceBundle ←------------- UserService
- DSO and BitstreamStorage ←------------- StorageService
Proposed Next Steps:

Replacement of Legacy Managers

- EventManager <-------------------------- EventService
- ConfigurationManager <---------------- ConfigurationService
- DatabaseManager <--------------------- DSpaceDataSource Service
- EPerson/ResourceBundle <-------------- UserService
- DSO and BitstreamStorage <------------ StorageService
- Search and Configurable Browse <------- SearchService
Proposed Next Steps:
Remove God Objects
Proposed Next Steps:
Remove God Objects

- Context is composite object
Proposed Next Steps:

Remove God Objects

- Context is composite object
- Gets passed around everywhere
Proposed Next Steps:

Remove God Objects

- Context is composite object
- Gets passed around everywhere
- Represents:

DSpace

Database Manager

Database Pool

Context

DB

cache

ID

Authentication Manager
Proposed Next Steps:

Remove God Objects

- Context is composite object
- Gets passed around everywhere
- Represents:
  - State, Identity, Transaction
Proposed Next Steps: Kernel as “Context Container”
Proposed Next Steps:
Liberate the Implementation
Proposed Next Steps:
Liberate the Implementation

- Remove Static Accessors allowing for proper API contracts and usage of Extensibility.
Proposed Next Steps:

Liberate the Implementation

- **Remove Static Accessors** allowing for proper API contracts and usage of Extensibility.
- **Decouple Initialization** of “StaticManagers” as Services into either core Spring, Guice or Application startup.
Proposed Next Steps:

Liberate the Implementation

- **Remove Static Accessors** allowing for proper API contracts and usage of Extensibility.

- **Decouple Initialization** of “StaticManagers” as Services into either core Spring, Guice or Application startup.

- **Enforce contracts and backward compatibility** as a community practice to assure reliable API + Services.
In Summary

- DSpace 2.0 is successful project to date.
In Summary

- DSpace 2.0 is successful project to date.
- Yet, will take multiple releases to integrate.
In Summary

- DSpace 2.0 is successful project to date.
- Yet, will take multiple releases to integrate.
- Work is incremental, projects need to be tractable.
In Summary

- DSpace 2.0 is successful project to date.
- Yet, will take multiple releases to integrate.
- Work is incremental, projects need to be tractable.
- Work needs to be kept close to the trunk
In Summary

- DSpace 2.0 is successful project to date.
- Yet, will take multiple releases to integrate.
- Work is incremental, projects need to be tractable.
- Work needs to be kept close to the trunk
- DSpace Services are here as the first step.
In Summary

- DSpace 2.0 is successful project to date.
- Yet, will take multiple releases to integrate.
- Work is incremental, projects need to be tractable.
- Work needs to be kept close to the trunk
- DSpace Services are here as the first step.
- Faster when we all collaborate in migration activities.
In Summary

- **DSpace 2.0 is successful** project to date.
- Yet, will take **multiple releases** to integrate.
- Work is **incremental**, projects need to be **tractable**.
- Work needs to be kept close to the **trunk**
- **DSpace Services** are here as the first step.
- Faster when we all **collaborate in migration** activities.
- Could always use a little more **“$upport”**
Special Thanks:

Ben Bosman
Art Lowel
Bradley McLean
Graham Triggs
Kevin Van de velde
Aaron Zeckoski
Supporting Organizations:

Mark Diggory
mdiggory@atmire.com
BACK-PORTING DSPACE 2.0: DSPACE SERVICES

Mark Diggory

DSUG 2009