Hudson Web Architecture

ORACLE®
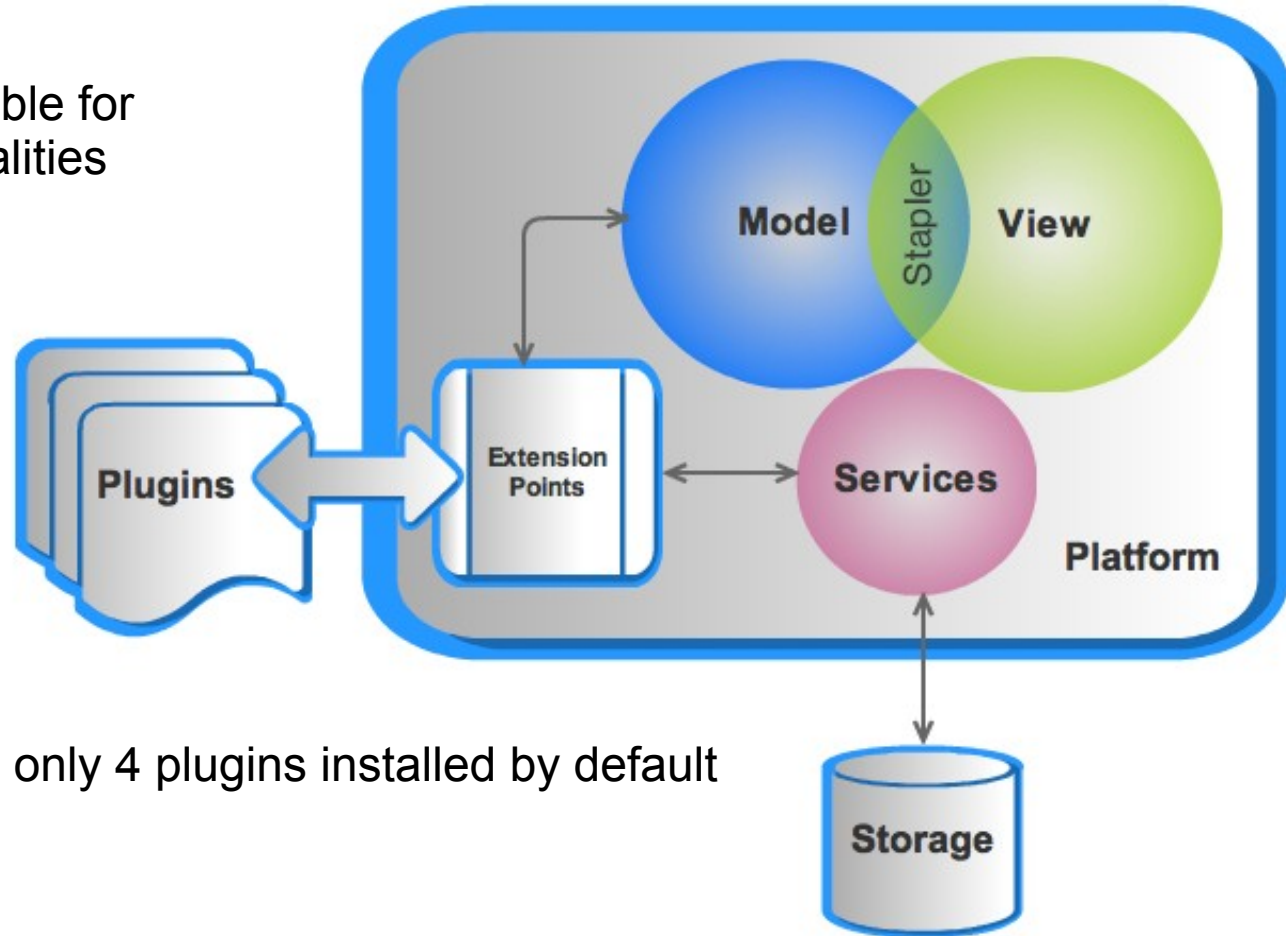
Winston Prakash

Architecture Overview

Hudson is an extendable Web Application. The basic platform is extended via plugins.

Plugins are responsible for
most of the functionalities
in Hudson

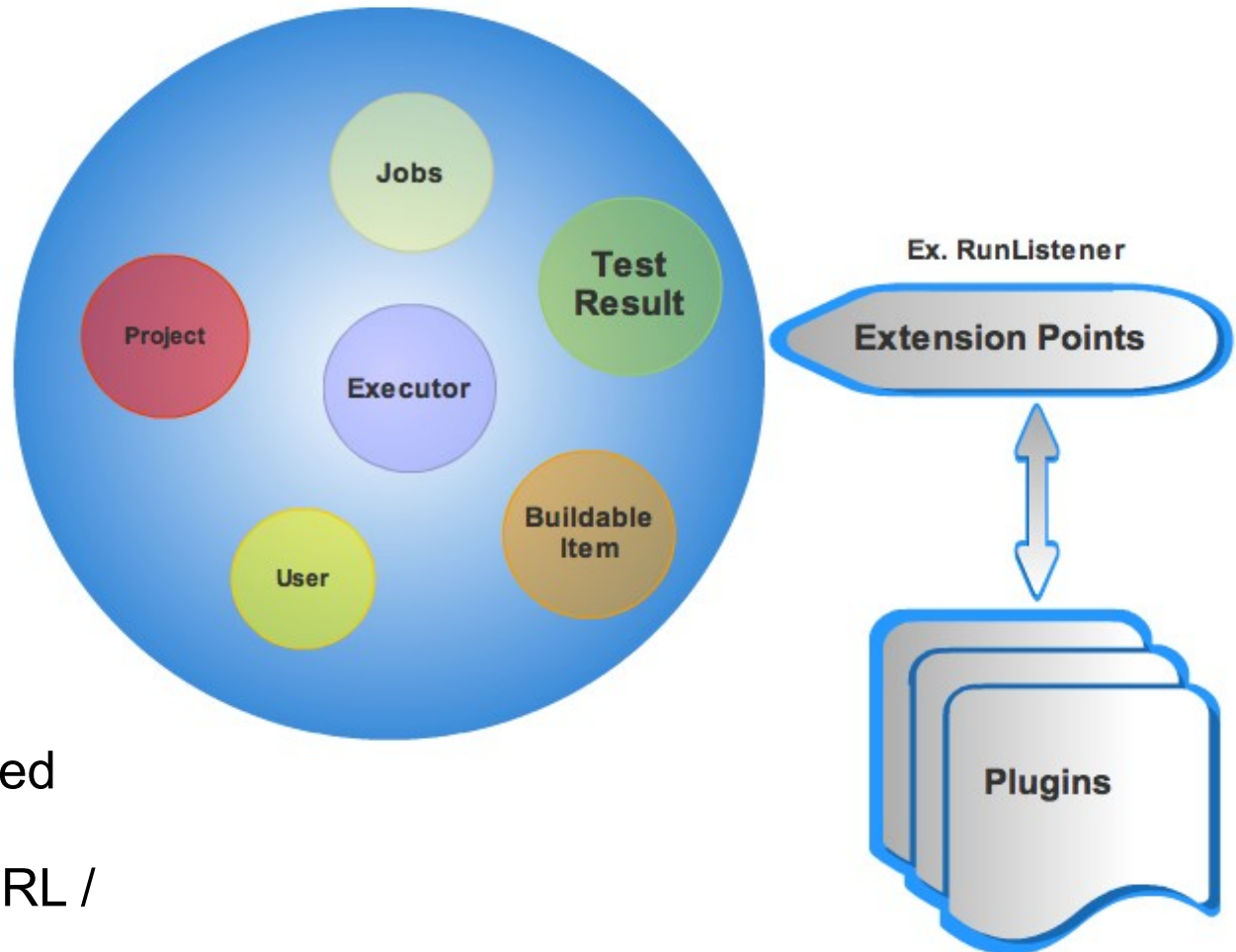Basic distribution has only 4 plugins installed by default

- CVS
- SVN
- Maven
- SSH

Model Objects

Model objects are building blocks of Hudson Platform. They hold the data and state of a Job/Run.
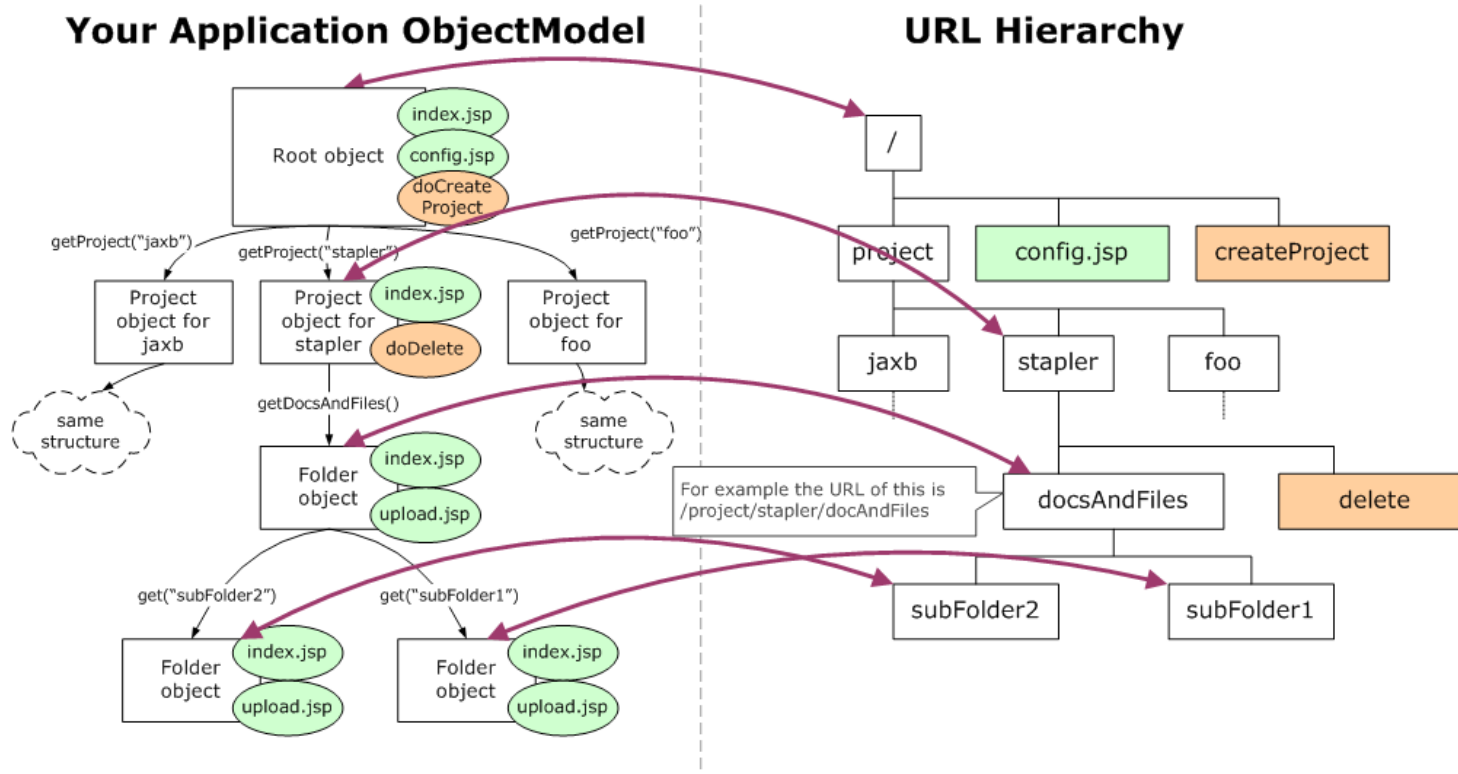
Each Model object is associated with an **URL**

The root object is called **Hudson** and it is associated with the URL /

Stapler – Stapling the URL with Model Objects

Model Objects are bound to URLs by using Stapler. The singleton Hudson instance is bound to the "/" URL, and the rest of the objects are bound according to their reachability from this root object.

Hudson Web Application Entry Point

Stapler is initialized in the Web.xml and specified as Servlet

```
<servlet>
  <servlet-name>Stapler</servlet-name>
  <servlet-class>org.kohsuke.stapler.Stapler</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>Stapler</servlet-name>
  <url-pattern>/</url-pattern>
</servlet-mapping>
```

```
<listener>
  <listener-class>hudson.WebAppMain</listener-class>
</listener>
```

→ Hudson Main Object gets Initialized

Specifying Hudson Root Object to Stapler

```
public class WebAppMain implements ServletContextListener {
    public void contextInitialized(ServletContextEvent event) {
        // Set Hudson Singleton as Root Object
        Stapler.setRoot(event,Hudston.getInstance());
    }
    public void contextDestroyed(ServletContextEvent event) {
        // Handle Hudson instance clean up
    }
}
```

The View

Hudson model objects have multiple "views" that are used to render HTML pages about each object. Hudson uses **Jelly** as the view technology
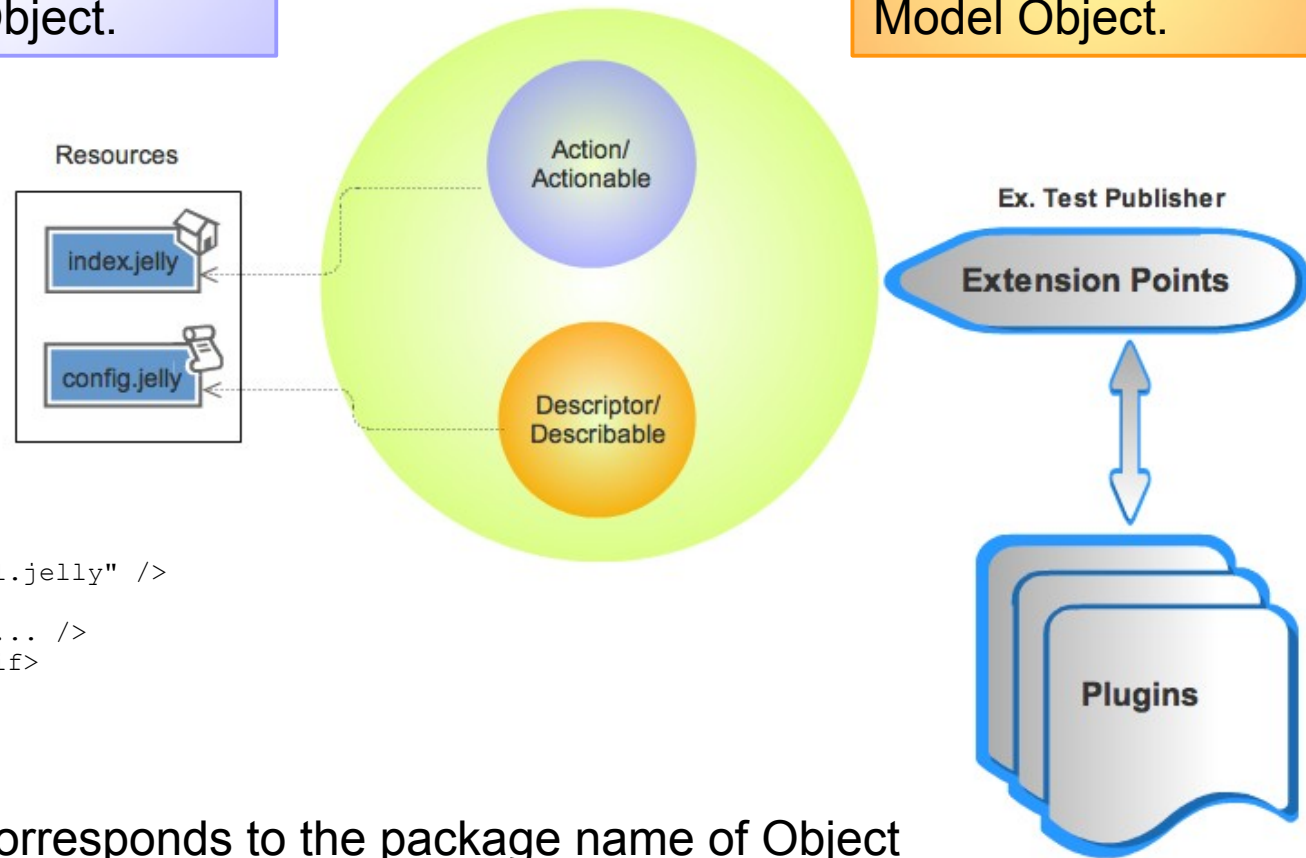
Actionable objects adds UI and information via **Actions** to Model Object.

Describable objects adds configuration UI via **Descriptor** to Model Object.

Jelly is the tag library used to define the UI.

Ex:

```
<l:layout .... >
    <st:include page="sidepanel.jelly" />
    <l:main-panel> ...
       <t:editableDescription ... />
        <j:if test="..."> </j:if>
    </l:main-panel>
</l:layout>
```

The UI name space corresponds to the package name of Object

Resources

index.jelly

config.jelly

Action/
Actionable

Descriptor/
Describable

Ex. Test Publisher

Extension Points

Plugins

ORACLE®

How Stapler Resolves Model Object and View?

Stapler resolves Model Object very similar to JSF Expression Resolver. it takes an object and URL, then evaluate the URL against the object. It repeats this process until it hits either a static resource, a view (such as JSP, Jelly, Groovy, etc.), or an action method.

Scenario: browser sends "POST /project/jaxb/testResult HTTP/1.1"

```
evaluate(<root object>, "/project/jaxb/testResult")
-> evaluate(<root object>.getProject("jaxb"), "/testResult")
-> evaluate(<jaxb project object>, "/testResult")
-> evaluate(<jaxb project object>.getTestResult())
```

Views

A Model Object can have associated "views", which are the inputs to template engines mainly used to render HTML. Views are placed as resources, organized by their class names.

For example, views for the class org.jvnet.hudson.project.testResult would be in the /org/jvnet/hudson/project/testResult/index.jelly

When they are executed, the variable "it" is set to the object for which the view is invoked. (The idea is that "it" works like "this" in Java.)

Stapler forward to "view" as

response.forward(this,"/org/jvnet/hudson/project/testResult/index.jelly",request);

and set the Model Object to the view via the variable "it"

Hudson Ajax Requests

Hudson Dashboard pages can have Ajax Requests. Ex: To show the status of Executing Builds. These requests are resolved by stapler similar to regular requests.

Build Queue

hudson_main_matrix

-> evaluate(<hudson.model.Hudson@50d4855d> :hudson.model.Hudson,"/ajaxBuildQueue")
-> evaluate(((StaplerProxy)<hudson.model.Hudson@50d4855d>).getTarget(),"/ajaxBuildQueue")
-> evaluate(<hudson.model.Hudson@50d4855d>.getDynamic("ajaxBuildQueue",...),"")
        hudson.model.Hudson@50d4855d.getDynamic("ajaxBuildQueue",...)==null. Back tracking.
-> evaluate(((StaplerFallback)<>).getStaplerFallback(),"/ajaxBuildQueue")
-> evaluate(<hudson.model.AllView@70a5d82a> :hudson.model.AllView,"/ajaxBuildQueue")
-> ajaxBuildQueue.jelly on <hudson.model.AllView@70a5d82a>

Build Executor Status

#	Status	
1	Building nighthacks-server #1022	☒
2	Idle	

-> evaluate(<hudson.model.Hudson@50d4855d> :hudson.model.Hudson,"/ajaxExecutors")
-> evaluate(((StaplerProxy)<hudson.model.Hudson@50d4855d>).getTarget(),"/ajaxExecutors")
-> evaluate(<hudson.model.Hudson@50d4855d>.getDynamic("ajaxExecutors",...),"")
        hudson.model.Hudson@50d4855d.getDynamic("ajaxExecutors",...)==null. Back tracking.
-> evaluate(((StaplerFallback)<hudson.model.Hudson@50d4855d>).getStaplerFallback(),"/ajaxExecutors")
-> evaluate(<hudson.model.AllView@70a5d82a> :hudson.model.AllView,"/ajaxExecutors")
-> ajaxExecutors.jelly on <hudson.model.AllView@70a5d82a>

ORACLE®

Hudson Services

The service objects of Hudson are Model Objects that are runnable.
Hudson executor runs these services to complete an execution.
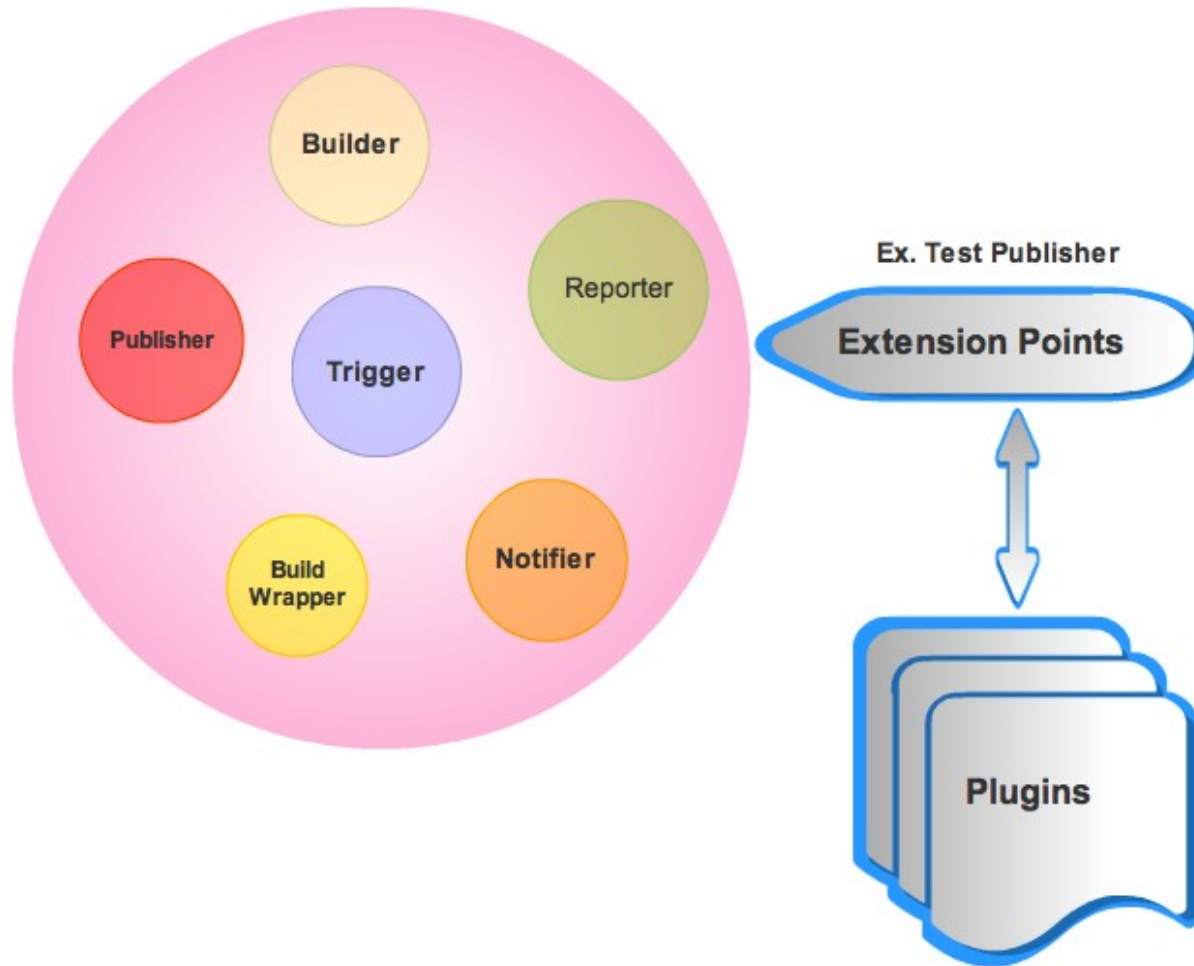
A Simplified Execution Scenario