Managing Diverse & Complex uPortal Deployments with Cernunnos

Andy Gherna & Drew Wills

JA-SIG Spring Conference, April 28, 2008

UNICON

Presenters

- Andy Gherna
  - University of Illinois at Urbana-Champaign
  - Worked with uPortal and portlets for 3 years
  - uPortal committer & Cernunnos committer
- Drew Wills
  - Unicon employee since 2000
  - Contributed to several uPortal implementations
    - California Polytechnic State University "Cal Poly“
    - California State University, Chico
    - University of Colorado System
    - Yale University
    - Johns Hopkins University
  - Launched Cernunnos February 2007

uPortal Deployment with Cernunnos

1. uPortal Deployment at U of I
2. The Cernunnos Project
3. Deployment Solved & Looking Ahead

uPortal Deployment at U of I

Problem: Multiple Campuses, Multiple Environments

- U of I faced a challenging deployment landscape
- 3 campuses
- 2 version control systems
  - SVN for source
  - CVS for deployed environment
- 4 environments
- many nodes

Solution 1: Ant

- Benefit: Ubiquity
  - you have it already
  - tasks make it easy (e.g. copy)
- Drawback: Scalability
  - change environment == change bulid process

Solution 2: Scripting

- Benefit: Flexibility
  - parameterize vs. modify
  - language features (python, groovy)
- Drawback: Scalability (again)
  - too many parameters
  - too much "plumbing code“
  - usability (who on my team can pick this up quickly?)

The Cernunnos Project

Software Solved

Cernunnos at a Glance

- Project Home Page:

  http://cernunnos.googlecode.com/

- Discussion Group:

  http://groups.google.com/group/cernunnos-discussion/

- Manual:

  http://cernunnos.googlecode.com/svn/manual/index.html

- Quick Facts:
  - 12k lines of source (code, comments, blanks)
  - 7 project members
  - 294 commits beginning 2007/02/15

What Is It?

- Cernunnos is hard to describe for the same reason it's interesting: *there's nothing else like it*
- I can't tell you what it is in a sound byte
- But it would go something like this:

  “Cernunnos is a form factor for code components designed to overcome the Problem of Implied Codemass.”

WTF?

- This description begs the question:

  “What is ‘Implied Codemass’ and why should I avoid it?”

- The reasons may sound "too good to be true“
  - You will do more in less time
  - You will produce higher quality deliverables
  - Your deliverables will get better with age (like wine)
  - You will learn a lot of cool things
  - You will teach others the cool things you know
  - You can turn in the same work again and again
  - You can turn in other people's work too
  - You can take it all with you to your next assignment

Implied Codemass

- Let's agree the *codemass* of a solution is the sum of all custom code
- *Implied codemass* is the portion of a solution that you're forced to write by architectural choices
- The real bulk, footprint, or mass of an individual choice -- whether you write it or download it -- includes the burden it places on components that interact with it

```
innate mass + implied mass = total mass
```

Why Implied Codemass?

- Industry-normal development practices lead to software systems that look like jigsaw puzzles
- Without pre-arranged guidelines for shape, components grow organically into shapes that reflect their purpose
- This process leads to systems made up of components that can only be combined in one way (like a jigsaw puzzle)

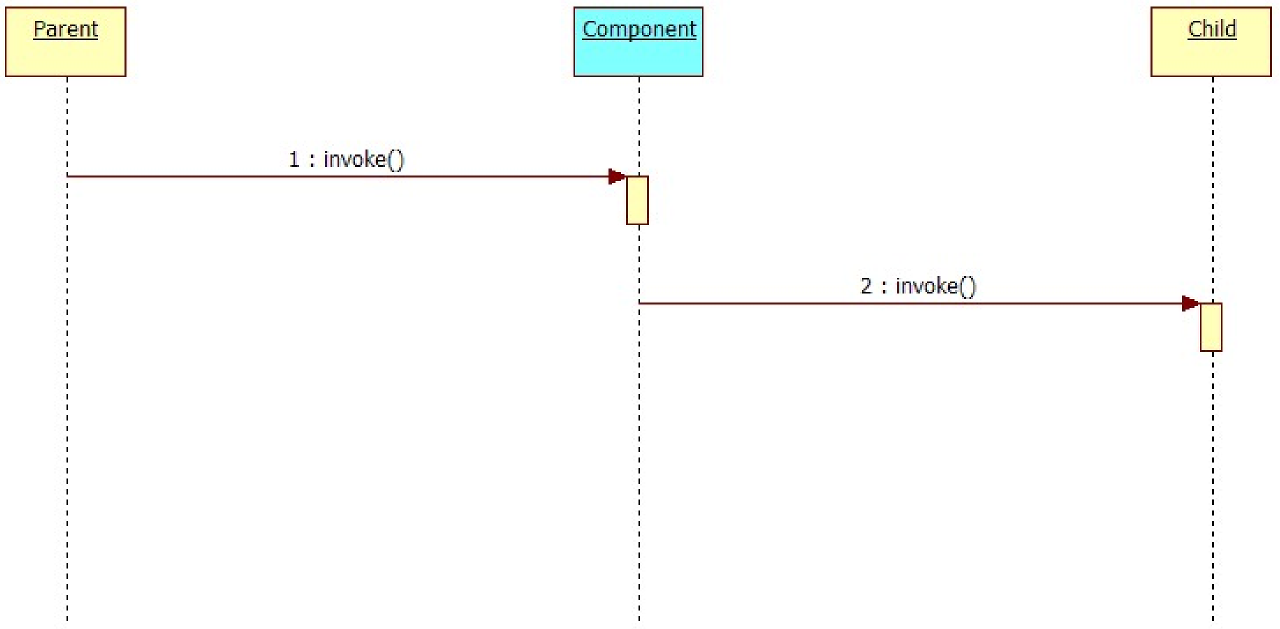
Example 1: Connect to and query JDBC

```
Connection conn = null;
PreparedStatement pstmt = null;
try {

    Class.forName("my.sql.Driver");
    conn = DriverManager.getConnection(url, user, passwd);
    pstmt = conn.prepareStatement("SELECT * FROM up_user");
    ResultSet rs = pstmt.executeQuery();
    /* do something useful with the results */

}
catch (ClassNotFoundException cnfe) { /* do something... */ }
catch (SQLException sqle) { /* do something... */ }
finally {
    pstmt.close();
    conn.close();
}
```

Example 2: Connect to and query LDAP

```
Hashtable env = new Hashtable(11);
env.put(Context.INITIAL_CONTEXT_FACTORY, "com.sun.jndi.ldap.LdapCtxFactory");
env.put(Context.PROVIDER_URL, "ldap://localhost:389/o=JNDITutorial");
env.put(Context.SECURITY_AUTHENTICATION, "simple");
env.put(Context.SECURITY_PRINCIPAL,
           "cn=S. User, ou=NewHires, o=JNDITutorial");
env.put(Context.SECURITY_CREDENTIALS, "mysecret");

try {

    DirContext ctx = new InitialDirContext(env);
    NamingEnumeration<SearchResult> rslt = ctx.search("",
                         "(objectclass=person)", null);
    /* do something useful with the results */

} catch (NamingException e) { /* do something... */ }
finally {
    ctx.close();
}
```

Hypothetical Sequence Diagram

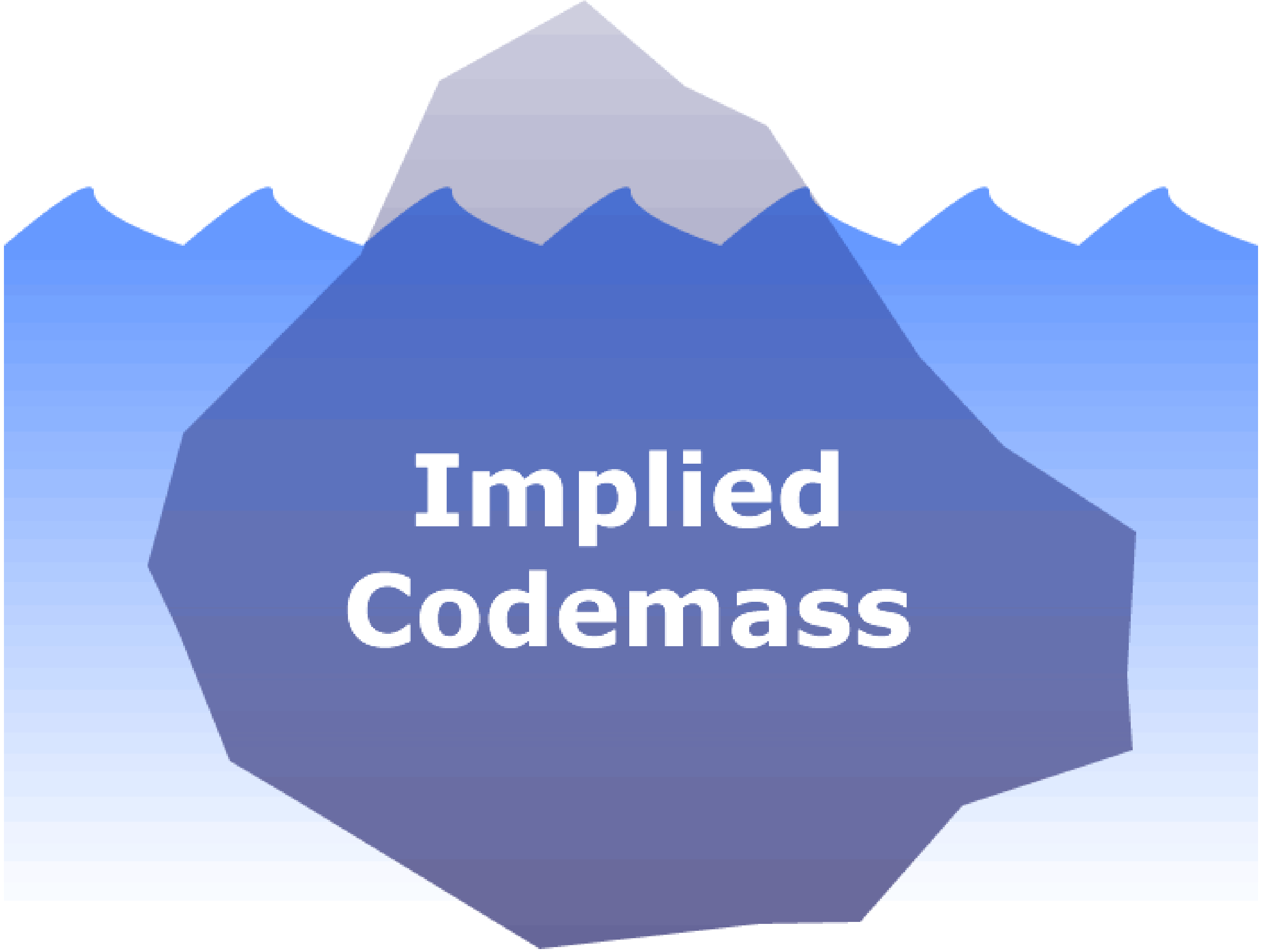


Why aren't the previous examples drop-in replacements for Component?

Implied Codemass (Cont.)

- Practices like these lead to *lots* of Implied Codemass
- Sometimes this extra bulk is overlooked
- In other cases it’s accepted without scrutiny because there's no perceived alternative
- In either case, the jigsaw puzzle paradim continues

Hidden Bulk



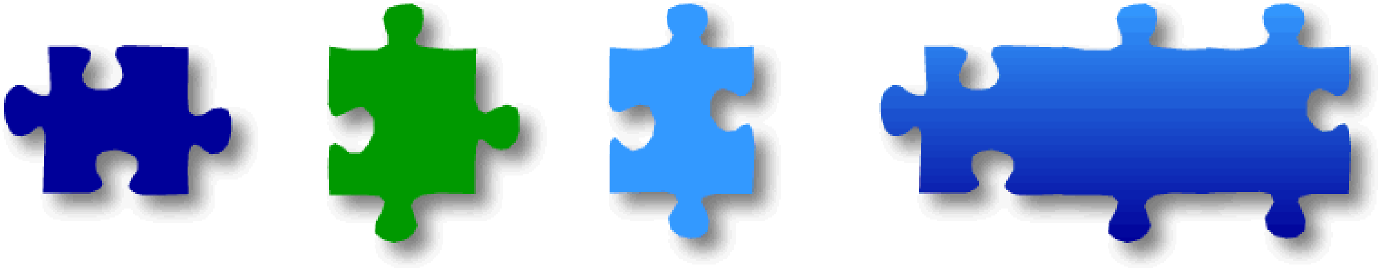
LEGOLAND®

“At LEGOLAND® everyone in the family becomes a hero.”

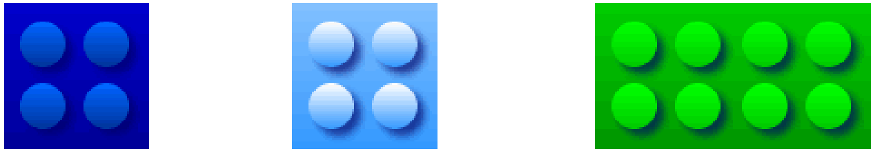
- Lego bricks are similar to puzzle pieces in many ways:
  - Intended for entertainment
  - Appeal to similar ages
  - Available in similar sizes and quantities
  - You combine them to make something bigger
- There's one essential difference:
  - with Lego bricks, the set of possible combinations is open-ended

GOAL: Combine A & B

A + C + B = AB

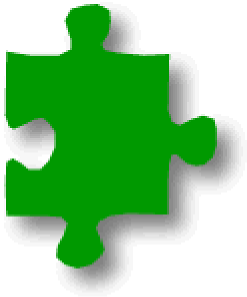


A + B = AB



Integration Paradigms (Cont.)

- The jigsaw puzzle paradigm requires the incremental mass of C



- The Lego Brick paradigm *requires no incremental mass*

Cernunnos Is Like LEGOs

- With Cernunnos, you don't have to make it possible for components to combine; you only have to tell them to do so
- When you introduce a new component to Cernunnos, it's instantly capable of combining with every other component -- past, present, and future

Cernunnos Example

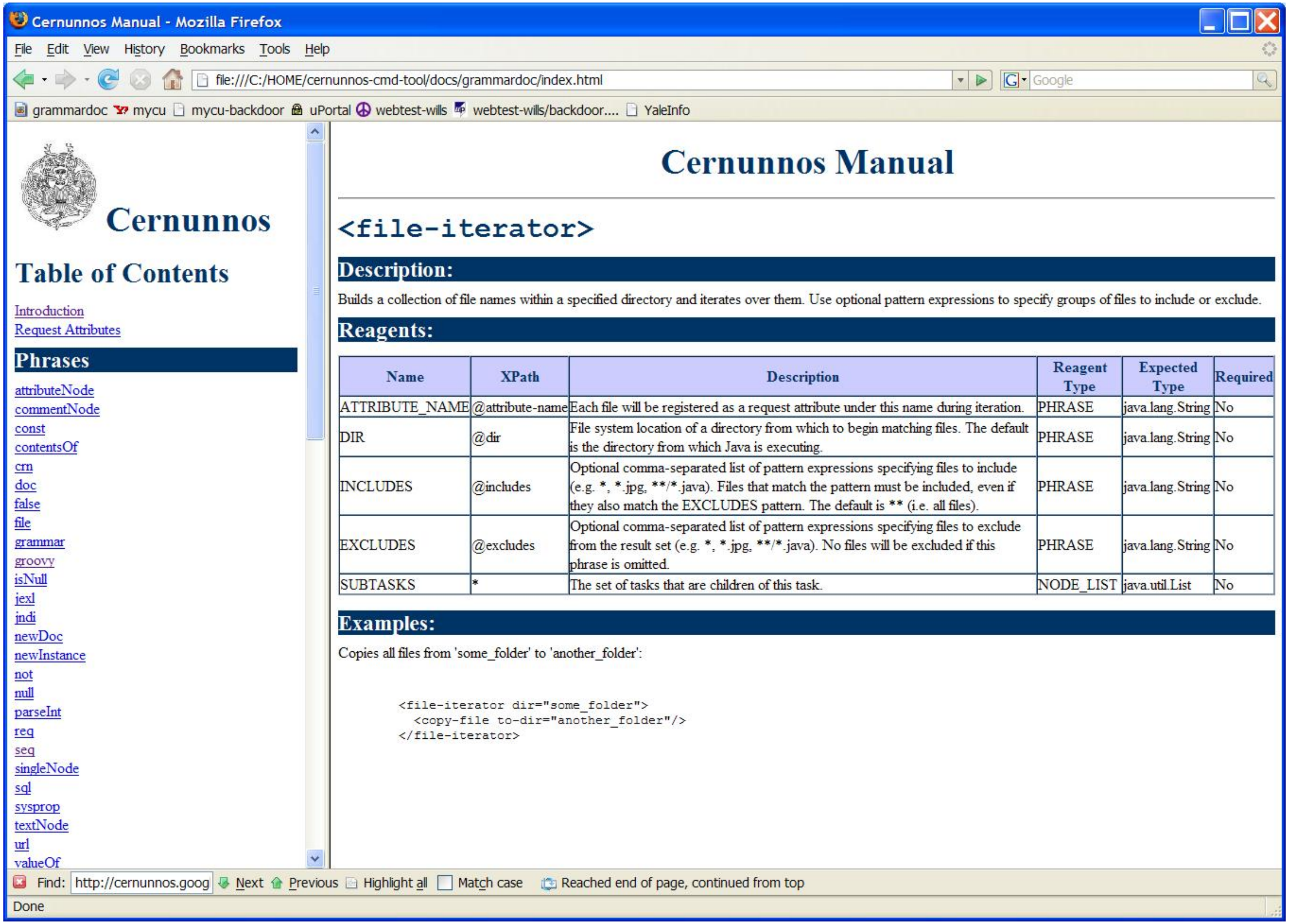
```
>crn find-entries-in-jars.crn proj\lib SAX
```

```
<!-- Find all JAR files in $1… -->
<file-iterator dir="${$1}" includes="**/*.jar">

  <!-- Print out each JAR name… -->
  <echo-ln>${Attributes.LOCATION}:</echo-ln>

  <!-- Print out entries that match $2… -->
  <archive-iterator>
    <with-attribute key="location" value="${Attributes.LOCATION}">
      <if test="${groovy(location.contains('${$2}'))}">
        <echo-ln prefix="&#009;">${location}</echo-ln>
      </if>
    </with-attribute>
  </archive-iterator>

  <echo-ln/>
</file-iterator>
```

Cernunnos Example (Cont.)

```
INFO:
**************************************************
** Invoking ScriptRunner.run(Task, TaskRequest)
** TaskRequest contains 3 elements
**   - $1=..\..\cernunnos-cmd-tool\lib
**   - Attributes.ORIGIN=file:/C:/HOME/find-entries-in-jars.crn
**   - $2=SAX
**************************************************

activation-1.1.jar:

commons-logging-1.1.jar:

commons-pool-1.2.jar:

jaxen-1.1.1.jar:
        org/jaxen/saxpath/SAXPathEventSource.class
        org/jaxen/saxpath/SAXPathException.class

jdom-1.0.jar:
        org/jdom/input/SAXBuilder.class
        org/jdom/input/SAXHandler.class
        org/jdom/output/SAXOutputter.class

…
```

Cernunnos Manual


Cernunnos Manual - Mozilla Firefox
File Edit View History Bookmarks Tools Help
file:///C:/HOME/cernunnos-cmd-tool/docs/grammardoc/index.html
Google
grammardoc mycu mycu-backdoor uPortal webtest-wills webtest-wills/backdoor.... YaleInfo
Find: http://cernunnos.goog Next Previous Highlight all Match case Reached end of page, continued from top
Done

Cernunnos

Table of Contents

Cernunnos Manual

<file-iterator>

Description:

Builds a collection of file names within a specified directory and iterates over them. Use optional pattern expressions to specify groups of files to include or exclude.

Reagents:

Name	XPath	Description	Reagent Type	Expected Type	Required
ATTRIBUTE_NAME	@attribute-name	Each file will be registered as a request attribute under this name during iteration.	PHRASE	java.lang.String	No
DIR	@dir	File system location of a directory from which to begin matching files. The default is the directory from which Java is executing.	PHRASE	java.lang.String	No
INCLUDES	@includes	Optional comma-separated list of pattern expressions specifying files to include (e.g. *, *.jpg, **/*.java). Files that match the pattern must be included, even if they also match the EXCLUDES pattern. The default is ** (i.e. all files).	PHRASE	java.lang.String	No
EXCLUDES	@excludes	Optional comma-separated list of pattern expressions specifying files to exclude from the result set (e.g. *, *.jpg, **/*.java). No files will be excluded if this phrase is omitted.	PHRASE	java.lang.String	No
SUBTASKS	*	The set of tasks that are children of this task.	NODE_LIST	java.util.List	No

Examples:

Copies all files from 'some_folder' to 'another_folder':

```
<file-iterator dir="some_folder">
  <copy-file to-dir="another_folder"/>
</file-iterator>
```

Do One Thing Well

- Cernunnos aims to make software less costly by taking on Implied Codemass
- Leave everything else to others
- Put good code in contact with other good code, get out of the way

Other Good Code

Groovy

Spring Framework

Java Community Process

JavaMail

<dom4j>

Spring LDAP

NekoHTML

commons DBCP

commons logging

Deployment Solved & Looking Ahead

Reconsidering Solution Design

- XML map of environment
  - edit the xml file, not the script
  - need a way to process the XML...
- Scripting languages: too much XML "plumbing code"

Solution 3: Cernunnos!

- Benefit: Scalability (at last!!!)
  - no "plumbing code“
  - XPath/node-iterator reads the XML map
- Benefit: Declarative XML
  - easy to pick up
  - develop useful applications quickly

deployment.xml

- Defines environments, applications, and app-servers

```
<deploy>
  <Environments>
    <Environment name="Development">

      <!-- The base directory for uPortal and portlets for each campus or machine type -->
      <Applications>
        <Application name="mydevuic"/>
        <Application name="axisdevuis"/>
      </Applications>

      <!-- The CATALINA_BASE directory for each cluster -> node -->
      <AppServers>
        <cluster name="mydevillinois">
          <config>
            <port type="SERVER">8111</port>
            <port type="HTTP_1_1">8112</port>
            <port type="AJP">8113</port>
          </config>
          <node>mydevillinois1_01</node>
          <node>mydevillinois2_01</node>
        </cluster>
      </AppServers>

    </Environment>

    [...]

  </Environments>
</deploy>
```

crn-deploy.xml

- Deploys a portlet application to the local directory of the deployable CVS module

```
<with-attribute key="DEPLOYMENT_XML" value="${file(deployment.xml)}">

  [...]

  <with-attribute key="JAR_FILE" value="${file(${req($1)})}">
    <with-attribute key="APP_NAME" value="${jexl(JAR_FILE.getName().replace('.war',''))}">

    <!-- Deploy the portlet app with the specified environment and cluster -->
    <if test="${jexl(APP_NAME.endsWith('-pac'))}">
      <node-iterator source="${doc(deployment.xml)}"
          xpath="//Environment[@name = '${req($3)}']/Applications/Application[@name = '${req($4)}']">
        <with-attribute key="$2"
            value="${req($2)}/${req($3)}/${valueOf(name(..))}/${valueOf(./@name)}">
          <crn location="deploy-portlet-app.xml"/>
        </with-attribute>
      </node-iterator>
    </if>

    [...]

    </with-attribute>
  </with-attribute>

</with-attribute>
```

Deployment Solved

- Cernunnos gave us what we were looking for
  - Consistency
  - Dependability
  - Confidence
- Looking ahead
  - additional deployment automation
  - subscriber services
  - uPortal data transfer

Questions?

Andy Gherna
aghernа@uiuc.edu
http://www.uiuc.edu/

Drew Wills
drew@unicon.net
http://www.unicon.net/blog/19/