

**Universiti Teknologi MARA**

**KnowledgePoint™: Developing a Seamless  
Integration for a Channel Focusing On Utilization of  
Java Classes/Servlets from Within OpenACS**



**Azrel Bin Ahamad**

Thesis submitted in fulfillment of the requirements for  
**Bachelor of Science (Hons)**  
**Information System Engineering**  
**Faculty of Information Technology And**  
**Quantitative Science**

APRIL 2005

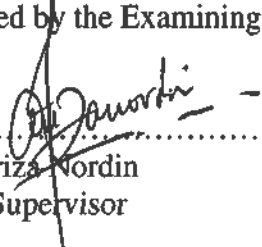
**KNOWLEDGEPOINT™: DEVELOPING A SEAMLESS  
INTEGRATION FOR A CHANNEL FOCUSING ON  
UTILIZATION OF JAVA CLASSES/SERVLETS FROM WITHIN  
OpenACS**

By

**AZREL BIN AHAMAD**

This thesis was prepared under the direction of thesis advisor, Prof. Madya Datin Dr. Noor Habibah binti Haji Arsyad, Department of System Science, and it has approved by the supervisor, Puan Ariza Nordin. It was submitted to the Faculty of Information Technology and Quantitative Science and was accepted in partial fulfillment of the requirements for the degree of Bachelor of Science

Approved by the Examining Committee:

  
.....  
Puan Ariza Nordin  
Thesis Supervisor

20<sup>th</sup> April 2005

## DECLARATION

I certify that this thesis and the research to which it refers are the product of my own work and that any ideas or quotation from the work of other people, published or otherwise are fully acknowledged in accordance with the standard referring practices of the discipline

.....  
AZREL BIN AHAMAD  
2003285347  
APRIL 20, 2005

## Acknowledgment

Praise to Allah S.W.T for giving me the opportunity to complete this dissertation. My highest gratitude goes to my supervisor, Puan Ariza Nordin, for her time, assistance, guidance and ideas throughout the project. Special appreciation is in order to Puan Suriyati, Encik Naim, Puan Natrah, and Puan Fauziah. And I would not be involves in this project without the initial job offer for SKP from Puan Amalina. Thus, thank you.

To Abah, Ma, Abg Eme, Hani, Harry, Fatul, Siti and Azlan, thank you for your relentless supports, encouragement, patience and doa all this time. It has been a very long journey. I love you all.

Lastly, I would like to use this opportunity to thank all my friends and classmates. Special thanks to Izudin, Titi Wangsa and Izwan Razif for your helps and contributions. Good luck and take care.

# TABLE OF CONTENTS

---

<b>APPROVAL</b>	ii
<b>DECLARATION</b>	iii
<b>ACKNOWLEDGEMENT</b>	iv
<b>TABLE OF CONTENTS</b>	v
<b>LIST OF TABLES</b>	viii
<b>LIST OF FIGURES</b>	ix
<b>ABSTRACT</b>	x
<b>1.0 INTRODUCTION</b>	
1.1 Research Background	1
1.2 Problem Statement	2
1.3 Objectives of Research	3
1.4 Significance of Research	3
<b>2.0 LITERATURE REVIEW</b>	
2.1 Introduction	4
2.2 Open Source Software	6
2.2.1 OpenACS	7
2.2.2 uPortal	9
2.2.2.1 uPortal Channels	10
2.3 Integration Approach using uPortal Channel	13
2.3.1 Custom Channel	13
2.3.2 WSRP Consumer Channel	15
2.3.3 Web Proxy Channel	20
2.3.4 XML Transformation Channel	22
2.4 KnowledgePoint™ Integration Implementation	24
2.5 Web Services and Components	26
2.5.1 Web Services	26
2.5.2 XML	28
2.5.3 XML Parser	28
2.5.3.1 DOM	29

2.5.3.2	SAX	29
2.5.4	SOAP	30
2.5.4.1	SOAP Message Structure	30
2.5.5	WSDL	31
2.5.6	UDDI	32
<b>3.0</b>	<b>RESEARCH METHODOLOGY AND APPROACH</b>	
3.1	Introduction	35
3.2	Research Project Phases	36
3.2.1	Theoretical Study	36
3.2.1.1	Literature Review	36
3.2.1.2	Developer Documentation for Open-Source	37
3.2.1.3	Review of Channel and Web Services Development Procedures, Guideline and Technival Reports	37
3.2.2	Exploratory Study	38
3.2.2.1	Focus Group interview	38
3.2.2.2	Planning design and method of documentation	38
3.2.2.3	Expert Network	38
3.2.3	Prototyping	39
3.3	Software Specification	39
3.4	Web Service Process	40
<b>4.0</b>	<b>CONSTRUCTION</b>	
4.1	Introduction	42
4.2	Web Service deployment at OpenACS and uPortal	42
<b>5.0</b>	<b>FINDINGS</b>	
5.1	Research Findings	45
5.2	Selection criteria for using Web Services	46
5.3	Web Services Tools	47
5.3.1	Apache Axis	47
5.3.2	Java WSDP	48
5.3.3	PHP NuSOAP	50
5.3.4	Perl SOAP::Lite	51

5.4	The Benefits of Web Services	52
5.5	Web Service Security Issues	53
6.0	<b>CONCLUSION</b>	55
	<b>REFERENCES</b>	56

## **List of Tables**

Table 2.1: Pros and cons of each uPortal channel type	12
Table 3.1: Software Specification for OpenACS installation	39
Table 3.2: Software Specification for uPortal installation	40



## List of Figures

Figure 2.1:	Open Source Application Development Framework	7
Figure 2.2:	uPortal and channels	11
Figure 2.3:	Custom uPortal channel connecting to OpenACS database using JDBC	13
Figure 2.4:	uPortal web service for WSRP Consumer channel	16
Figure 2.5:	OpenACS server acting as uPortal WSRP web service	18
Figure 2.6:	uPortal Web Proxy channel connecting to .LRN server using HTTP requests	22
Figure 2.7:	uPortal XML Transformation channel requesting XML document and XSL stylesheets	23
Figure 2.8:	Type of Integration for Portal Architecture	25
Figure 2.9:	Web Services Programming Stack	27
Figure 3.1:	Overview of Research Methodology and Approach	36
Figure 3.2:	Web Services Architecture	41

## **Abstract**

KnowledgePoint™ is the Knowledge Portal architecture for the Systems Science Department of the Faculty of Information Technology and Quantitative Science (FTMSK). Open source initiative is making its root here at FTMSK with an ongoing development of the knowledge repository and knowledge map layer within OpenACS web application framework. OpenACS is an open-source software for web application framework that build on Tcl scripting language. There is also other online system in research such as E-Portfolio to cater for lecturers' information. An open-source portal framework that utilizes Java, uPortal framework will be use as a single point entry for the KnowledgePoint™. This paper is aims at finding a suitable mechanism to integrate the OpenACS web application framework with the uPortal framework. One way to do it is by using web services.

# CHAPTER 1

## INTRODUCTION

### 1.1 Research Background

KnowledgePoint™ is the architecture of the Knowledge Portal for the Systems Science Department of the Faculty of Information Technology and Quantitative Science (FTMSK) knowledge management project. Currently there are ongoing developments of the knowledge repository and knowledge map layer within OpenACS web application framework.

Knowledge portals are single-point access software systems intended to provide easy and timely access to information and to support communities of knowledge workers who share common goals. The vision for knowledge portals is to support knowledge workers in their “gathering of information relevant to a task, organiz[ing] it, search[ing] it, and analyz[ing] it, sythesiz[ing] solutions with respect to specific task goals, and then shar[ing] and distribut[ing] what has been learned to other knowledge workers” (Mack, Ravin, and Byrd, 2001). Knowledge portal technologies are typically associated with a set of administrative tasks, requiring the exercise of various kinds of expertise (Mack, Ravin, and Byrd, 2001). By using application enabling middleware or web services it will allow results to be merged and manipulated in searches across multiple heterogeneous databases.

The OpenACS (Open Architecture Community System) application framework is an open source system that was initiated by the ArsDigita Corporation. It is based on the former Arsdigita Community System (ACS) that was first initiated in 1997 by Phillip Greenspun and others as a form of solution to the common problem of “creating scalable web community systems” as well as reaching a compromising level of software reuse across website

developments in communities that have similar business requirements (Calvo and Peterson, 2002).

In general, an application framework can be defined as a “reusable, ‘semi-complete’ application that can be specialized to produce custom applications” (Calvo and Peterson, 2002), which essentially provides software developers with a number of benefits consisting of modularity, reusability, extensibility and inversion of control. The term “web application framework” relates to these framework architectures that are designed specifically to meet the needs of a particular web application domain.

A channel that can utilize Java classes/servlets from within OpenACS web application framework will open a possibility of integrating OpenACS application with uPortal, an open source portal framework. uPortal is an open-standard effort using Java, XML, JSP and J2EE. It is a collaborative development project with the effort shared among several of the Java in Administration Special Interest Group (JA-SIG) member institutions of higher-education. It is a portal framework that has set of technical specifications, and software. The framework provides a J2EE portal server (container) and well-defined interfaces that will permit individual institutions to customize the institutional portal by plugging in components in a well-defined and usable manner. Integration with existing campus infrastructure can be simplified by the platform.

## **1.2 Problem Statement**

uPortal framework is gaining a lot of interest and support from many higher education information technology communities around the world because of its open source nature and it adheres to open standards, as well as its feature-rich with nice presentation layer for various information resources on an organizational level. The primary function of the framework is to provide an efficient and flexible engine for assembling a presentation. However, uPortal framework does not have course management facilities that are essential for e-learning. On the other hand, OpenACS framework through its e-learning infrastructure, dotLRN, offers

complete course and community management infrastructure that is scalable, robust, extendable and complies to open standards. This is one of the reason OpenACS is selected as KnowledgePoint web application framework. The software, which is currently used to deliver online courses and collaboration tools at a number of major universities, implements an object oriented data model on a relational database. The issues here are to find an optimum solution for integrating both frameworks since both frameworks are based on different platform, that are TCL and Java, and also both frameworks have their own authentication and authorization systems.

### **1.3 Objectives of the Research**

The objectives of this research are:

- i. To setup OpenACS working environment.
- ii. To find an optimum way to implement a channel or API prototype that will cater the use of Java classes/servlets from within OpenACS platform.
- iii. To deploy an OpenACS application through servlets channel into uPortal framework seamlessly.

### **1.4 Significance of Research**

- i. Initiating an open source e-learning infrastructure at FTMSK.
- ii. Explore and study the implementations of web services as a method to provide access to the information and services through a single portal interface for students, lecturers and staffs.

## CHAPTER 2

### LITERATURE REVIEW

#### 2.1 Introduction

Since the introduction of the World Wide Web and increasing global access to the Internet, educational institutions of all types have experimented with what has come to be known as e-learning. Sun Microsystems explains e-Learning as the ability to use the Internet, networking computing and other electronic technologies to facilitate, measure, and manage learning (Sun Microsystem,2003). What's needed today is a more modular approach, building on a solid hardware, software, networking and development foundation. It must provide a uniform approach to the production and reuse of learning objects. It is important that the existence heterogeneous computing and applications environments to be able to work together seamlessly. One important factor is that e-learning framework should provide an easy and reliable Internet-based method to create and access learning content to different type of users in the campus. This focus on the learner, not the technology, is the key to any successful e-learning implementation.

Nowadays there are even open-source and freely available e-learning framework and portal framework such as OpenACS web application framework, the Apache Project's JetSpeed (Jakarta Project JetSpeed Portal) and JA-SIG's uPortal which offers the opportunity to create personalized and dynamically-generated online experiences using technologies of that commonly deployed in the commercial internet world. This open-source software are increasingly popular in North America and Europe due to their similar nature of knowledge sharing in the academia and open-source communities as well as the ability to do customization on the software, no risk of vendor lock-ins and able to maintain the value of training processes and materials. (Calvo, R.A, Ghiglione, E. and Ellis, R.A, 2003)

An application framework can be defined as a “reusable, ‘semi-complete’ application that can be specialized to produce custom applications“(Calvo and Peterson, 2002), which basically provides software developers with a number of benefits consisting of modularity, reusability, extensibility and inversion of control. The term web application framework relates to the framework architectures that are designed specifically to meet the needs of a particular web application domain (Calvo and Peterson, 2002).

OpenACS allows website developers to rapidly implement modular and extensible web applications. Its also allow the developers to achieve significant levels of software reuse by utilizing the features of the core OACS packages. In the ongoing KnowledgePoint™ project, uPortal framework will reside on top of the OpenACS web application framework. Reasons are to use uPortal to provide a seamless presentation of information that comes from multiple external services (known as channel) through the portal user interface, to create personalization and single sign on capability for the Information Systems (Brusilovsky, 2004). The aggregation of information is to integrate content from different sources within a web page. The primary function of the uPortal framework is to provide efficient and flexible engine for assembling a presentation.

This chapter’s purpose is to make the reader familiar with the technologies and theories that are relevant for this paper. This chapter is not meant to be complete reference or tutorial of the presented technologies or theories. Instead after reading this chapter the reader of this paper should know the relationships of used methods and be familiar enough with them to be able to read the rest of this paper.

## 2.2 Open Source Software

Open-source software (OSS) is any computer software distributed under a license which allows users to modify and redistribute the software freely (Open Source Initiative, 2005). Open-source software is required to have its source code open, unrestricted and freely available by downloading it from internet. The 'open' in open source software is intended in the philosophical sense of 'open or free speech' rather than as a free or no cost product. When the source code of a program is freely available, a worldwide community of Open Source developers can participate in peer review. When programmers can read, redistribute, and modify the source code for a piece of software, the software evolves. People improve it, people adapt it, people fix bug (Ayala et al, 2002). And this can happen at a speed that, if one is used to the slow pace of conventional software development, seems astonishing. The rapid evolutionary process produces better software than the traditional closed model, in which a very few programmers can see the source and everybody else must blindly use an opaque block of bits

Open-source licenses may have additional restrictions, such as a requirement to preserve the authors' names and copyright statement in the code. Open-source software is way of writing, selling and supplying software that help software get cheaper and more flexible. Example of success open source projects includes Linux operating system, OpenOffice.org, PostgreSQL, and Apache.

Meanwhile, proprietary software is applies to any software that is non-free or partially free in which the user does not control what it does or cannot study or edit the code. The modification, use and redistribution is prohibited, or requires express permission from the originator. Generally, it means that the customer will only get a binary version of the computer program they licensed and no copy of the program's source code, rendering modifications to the software practically impossible from the technical side, because the usual way to modify a program is to edit its source code and then compile it. The company or owner of the proprietary software holds the exclusive copyrights on a piece of software. (Wikipedia, 2005)



## 2.2.1 OpenACS

OpenACS is an open source and communities web application framework which enables web developers to rapidly implement modular and extensible web applications by utilizing available features of the core OpenACS packages in order to achieve significant levels of software use (Calvo and Peterson, 2002). OpenACS is downloadable set of web application software, which is under General Public License (GPL) meaning it is free for use and modification. Some of the OpenACS built-in applications are: Calendar, Content Management System, Portal System, Forum, E-Commerce application, photo album and news. It is mainly implemented via TCL, as a group of applications that interact with AOLServer and a RDBMS (so far only PostgreSQL and Oracle) and can be divided into a number of core packages, services and applications.

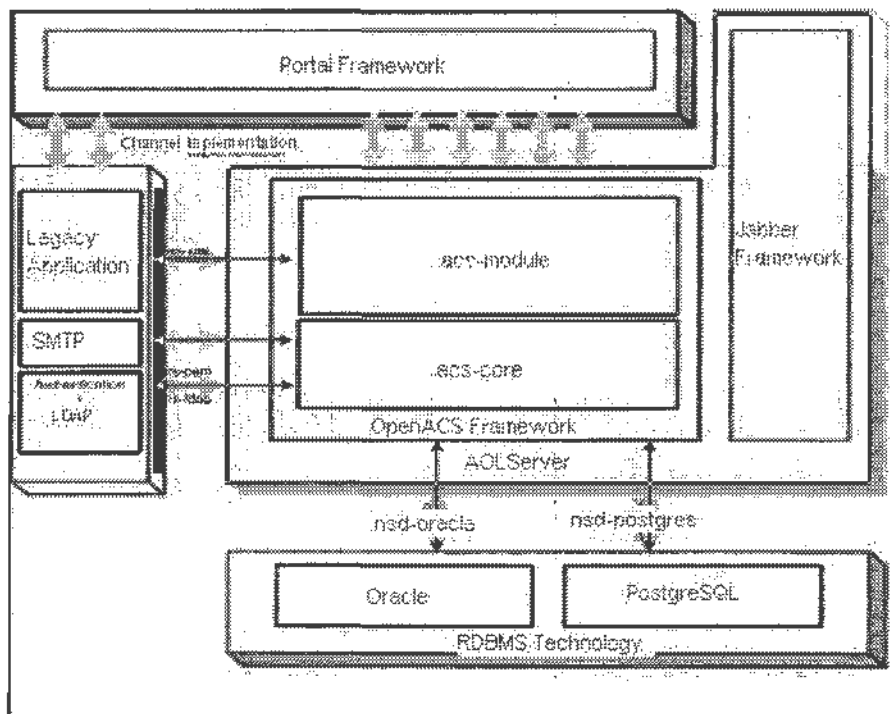


Figure 2.1 Open Source Application Development Framework  
(Source: Md. Nor et al, 2004)

The main base layer is the “acs-core” consists of the core packages required for OpenACS to function (Figure 2.1), which include a kernel, package management, administration utilities (users, sitemap and other common business objects), a template system that cleanly separates the business logic from the presentation layer, and a backend mail and messaging system (Calvo, Ghiglione and Ellis, 2003).

OpenACS services, such as the events package, are implementations of reusable logic that does not have user interface and not application dependant. Application packages are used to interact with the users. Application packages usually coincide with the service packages to provide the functionality for the users. These packages have a user interface and their code is clearly divided into application and presentation logic (Calvo, Ghiglione and Ellis, 2003).

In OpenACS every piece of contents are registered in a content repository that standardizes the way applications access and manage the information. This additional layer of complexity pays off by making possible the reusability of all content management functionalities such as permissions, formatting or other functionalities that may have not been included at the time of designing an application. A service created for the content repository, such as information retrieval or automatic document classification system, can automatically be used by all applications in the system. With a different architecture, the service would need to be re-implemented or adapted for each single application, making it aware of its internal representation (Calvo and Peterson, 2002).

Each service and application requires a data model that will be integrated into the framework. The integration of these data models enables reusability and a number of functionalities. Since the RDBMS does not implement Object Oriented (OO) functionalities to increase the reusability of the framework, and the framework has object oriented architecture, they must be added ad-hoc by the framework. The OpenACS objects have their attributes stored in tables and a set of methods defined as

PL/SQL packages. These packages hold the procedures that make the programming interface for the data model (Calvo, Ghiglione and Ellis, 2003).

The idea behind the Object Oriented architecture of OpenACS is that each piece of information that might be reusable should be an object. Since RDBMS are inherently not OO, data structures are integrated into the framework by being added to an `acs_objects` table that keeps track of every OpenACS object. Another table called `acs_object`, defines the standard attributes stored on every object, including a system wide unique ID, an object type and auditing columns. The concept of OpenACS object types is equivalent to classes in Object Oriented programming languages. Since OpenACS are using a RDBMS, additional work must be performed (Adida, 2003).

### **2.2.2 uPortal**

uPortal is an open source portal framework for an integrated delivery of content gathered from different sources of information. uPortal framework consists of a set of Java classes, XML data files, XSL stylesheets, and documents for producing customized campus portals.

A framework is a reusable design, an infrastructure that supports a coherent architectural model, enabling developers to concentrate on providing customized solutions. The uPortal framework delivers the common functionality that every portal needs, leaving developers to implement features that are specific to individual institutions by plugging in components in a well defined and usable manner (Sun Microsystems, 2002). The underlying model is based on a management structure of publish and subscribe, in which all members of the campus community both provide and consume content of interest to the community and to themselves. This structure includes enterprise applications, channels, publications, and links. The portal specification provides single sign-on plug-and-play, providing both ease of use and the ability to implement single sign-on in a way appropriate to the situation.

One of the great advantages to the uPortal design is the fact that the layout data, the structure and the look of the final layout are all independent of each other by using channels. Therefore, the information that is taken out of the database can be transformed by different XSL documents, rendering the same layout data into a wide variety of final views (JA-SIG, 2003). Given a set of information sources (channels), and layout on how to arrange and frame them (stylesheets), uPortal framework able to coordinate the compilation of the final document.

### 2.2.2.1 uPortal channel

Channels are the unit source of information for the uPortal and can also be defines as an area on website that contains information or contents of specific topic. Channel also an application that gathers data from a back-end data source and combines it with display templates so that the content can be mapped to a portal for viewing. The framework organizes channel behavior to produce a coordinated output of the content provided by them (JA-SIG, 2003). uPortal framework provides channels with the means to achieve their goals, but does not try to enforce how things are done. There are 8 types of channel support and can be implement by uPortal. Each different channel type has settings used to describe the location and types of resources needed for it (Barret, 2002). These are specialized to the channel type (Barret, 2002):

1. **Applet** – Renders a Java Applet in a uPortal Channel.
2. **Image** – A container for rendering media, not just graphic images. It supports any type of media that can be displayed or played through the browser including graphics, flash objects, movies and audio.
3. **Inline Frame** – Similar to the Image Channel, the Inline Frame is a container for displaying HTML content. The content in an Inline

Frame channel does not go through the uPortal framework, so it does not have to be well formed XHTML.

4. **RSS** – Content rendered in RSS (Rich Site Summary) format
5. **Web Proxy** – Similar to the Inline Frame Channel Type in that it is a container from HTML content. However, it requires well formed XHTML and can also render an XML application.
6. **WSRP Consumer** – Uses SOAP to communicate with and present the contents of a channel living in a remote instance of uPortal
7. **XML Transformation** – Transforms an XML document using a set of XSL stylesheets specified by uPortal
8. **Custom** – Customized channel content with all required resources provided by the writer of the channel.

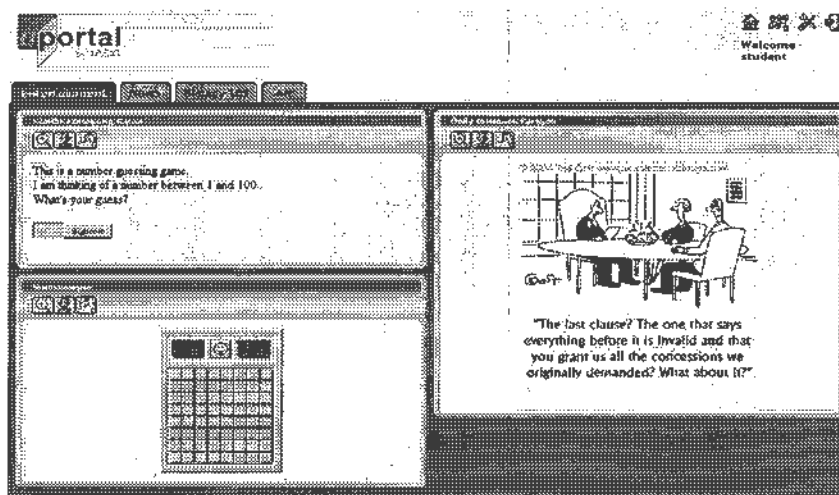


Figure 2.2: uPortal and channels (Source: JA-SIG, 2003)

Table 2.1 below list few pros and cons for each channel type available in uPortal (Barret, 2002). Information from the table is useful and important because it can be use to determine the optimal approaches to do an integration between OpenACS and uPortal framework.

Channel Type	Pros	Cons
<b>Image</b>	Easy to define.	Limited usage for displaying images.
<b>Inline Frame</b>	The easiest way to define a channel within uPortal. The remote site is displayed within a channel on the portal as an Inline Frame.	Not all browsers support inline frames. It cannot receive parameters and/or credentials from uPortal and the state of the channel is not preserved while interacting with other areas of uPortal.
<b>RSS</b>	"Look" like native portal channels. Can provide a link to the "host" of the RSS channel and links to points of interest and an optional text input field.	Provide no internal navigation mechanisms. Control over the "look and feel" is suppressed by the specification.
<b>Web Proxy</b>	Can be deploys at any remote web server. Control of layout and behavior is given to the remote HTML tagger. Easy to "portalized" any web application with little or no modification at all. Performance and availability of uPortal is not affected by performance and availability of remote web server.	Requires a well formed HTML. A sophisticated channel will require Web Proxy to rewrite most all URL references within the HTML document.
<b>WSRP Consumer</b>	Allows a complete interactive channel to be offered in multiple portals while maintenance of the channel is only done in one portal. Easy to add interactive content. Provides an easy way to preview a channel from another portal.	A sufficient authentication strategy is not yet in place.
<b>Custom</b>	Can usually be shared with other organizations utilizing uPortal, Easily make use of features of uPortal, and can "call" other custom channels within uPortal to easily expand their capabilities. Produce the most efficient channels	The most difficult to develop and poorly designed channel can significant impact the performance of uPortal. Can affect the availability of uPortal by causing JVM core crashes.
<b>Applet</b>	Defining an Applet channel is actually very easy. Applets, as are very powerful. Provide a distribution solution by requiring browsers to pull them from a server. Much more feature rich GUI.	Very complex to implement and not widely use in the industry.

Table 2.1: Summary of Pros and Cons for each uPortal Channel Types

## 2.3 Integration Approach using uPortal channels

Four out of eight uPortal channels discussed in Section 2.2.2.1 offer the potential for integrating uPortal with OpenACS, which are: Custom channel, WSRP consumer channel, Web Proxy channel and XML transformation channel. The rest of this section will be used to discussing these remaining channel implementations in the context of integration with OpenACS.

### 2.3.1 Custom channel

Custom channel offer the most flexibility for integrating OpenACS with uPortal, and they also require the most effort of the integration approaches that will be discussed in this report. A custom channel is a specially developed uPortal application, and the development process is similar to developing servlets in a J2EE application. With regards to integration with OpenACS, this development approach offers the potential for creating a completely customized application that uses OpenACS as a data-source, either through the use of JDBC or some type of web-service that provides the back-end datasource for the uPortal custom channel (Wickstrom, 2003). The JDBC approach is shown in Figure 2.3 below

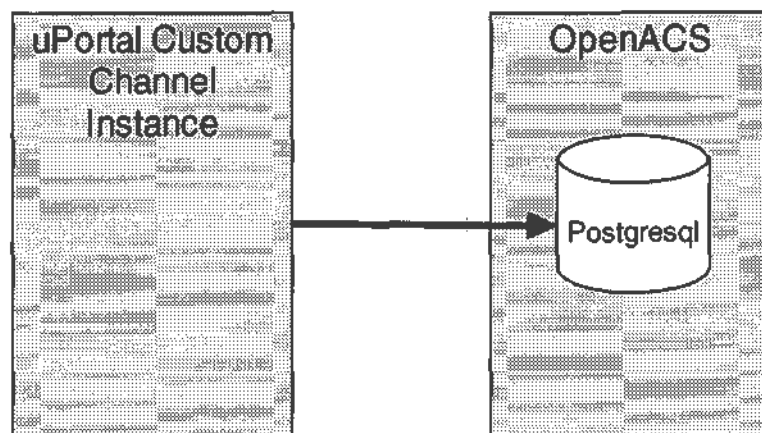


Figure 2.3: Custom uPortal channel connecting to OpenACS database using JDBC

(Source: Wickstrom, 2003)

uPortal makes use of JNDI services to provide access to database connections, and it is only necessary to change configuration files to enable JDBC connections to either database. While creating a custom channel for uPortal that uses JDBC connections to connect to the OpenACS database offers significant flexibility for development, it also shifts most of the development effort from OpenACS to uPortal. (Wickstrom, 2003)

This method requires developer to be well versed in java web-server development tools and strategies and they would also need to have intimate knowledge of OpenACS internals. An OpenACS developer would need the following skills in addition to their knowledge of OpenACS:

1. JDBC use with Oracle or Postgresql
2. JNDI for retrieving DB connections or LDAP connections
3. Servlet programming
4. XSL - XSL templates are much more complex than OpenACS templates, so the time to become productive using them would be much longer
5. SAX and DOM xml parsers
6. Ant-based build process
7. Servlet setup and configuration

From a high-level, a custom channel developed using this approach is essentially an OpenACS package developed in Java that supports a uPortal IChannel interface for interacting with the uPortal framework. The IChannel interface that follows consists of the methods that must be implemented to support any uPortal channel. The interface consists of the following five methods (JA-SIG, 2002):

1. *void setStaticData(ChannelStaticData sd)* - passes static data to the channel when it is instantiated