1. Ongoing Project Background

As part of the course on-going project, you will develop an Enterprise application of your choice and deploy it on top of various types of application server environments. You will also enrich your application, and tune its operational environment as allowed by the underlying application server facilities. You will need to develop and deploy your application as a set of application-level services and common facilities on top of various application server frameworks suggested as part of the homework specifications. Your applications should keep the processing, rendering, querying, secure messaging, and persistence aspects of your application well componentized to help study the migration of these capabilities across various types of application server platforms. As you transition your Enterprise application from one platform to another, you will study migration and interoperability facilities that leverage off of the cutting edge concepts subsumed by modern Model Driven Architectures (MDA) and Pattern Driven Architectures (PDAs).

You should follow the "real-life" application architecture mapping approach suggested in the Session 2 slide set. In real-life projects, the application, application infrastructure, and technology infrastructure are usually analyzed, designed, and developed in parallel. You should always select a technology infrastructure (e.g., Windows XP/Vista or Solaris on i5.nyu.edu, and Sun’s JDK 6.x) compatible with the application infrastructure set forth by the corresponding homework.

2. Introduction to Traditional Application Server Technology

The first generation application servers were spawned from the requirement to improve upon basic CGI connections between a database and a Web server. To relate to this initial requirement, this first homework studies a simple Application Server configuration based on an extension of basic CGI technology.

The simple Application Server is made up of the following components (see also attached diagrams)

1. A small CGI client program which could be written in any language

2. An Application Server which is multithreaded and is written in Java
3. Templates which contain the presentation logic of the screens. By keeping the presentation logic detached from the actual code, there is no need to recompile the Application Server to create new screens allowing for rapid development and maintenance. In addition, these templates can contain logic for remote method calls or queries to an underlying database.

2.1. Software Infrastructure Provided:

*Programming environment*

2. Apache Web Server archive to be downloaded from [http://www.apache.org](http://www.apache.org). Other web servers can be used as alternatives (e.g., IIS, etc.)
3. Browser (e.g., Internet Explorer)
4. JSE 6.x
5. A Java IDE of your choice
6. An OOAD environment of your choice (e.g., Rational Rhapsody Modeler, Poseidon, etc.)
7. Other development tools best suited for legacy application server development environments (as identified in homework #1a)

*Programs*

1. AppServer.java
2. cgimodule.pl

*Templates* (these files must be placed in the same directory where AppServer has been started)

1. Login.html
2. Main.html
3. Screen1.html
4. Screen2.html

*Additional Sample Applications*

1. JavaCGI
2. PatientProfileDB

2.2. Installing and Running the Homework Software Infrastructure:

1. Install Perl
2. Run the AppServer
   ✓ Make sure to place all templates in the same directory as AppServer
   ✓ Run the AppServer as follows: AppServer port#
3. Run the cgimodule
   ✓ Make sure to place the cgimodule.pl in the Web server cgi-bin directory
   ✓ Make changes to the script so that the correct host and port are set
   ✓ Run the cgimodule as follows from your web browser:
     http://localhost/cgi_bin/cgimodule.pl?NextTemplate=login.html

3. Questions

1. Preparation phase:
a. Select a technology infrastructure that is compatible with the application infrastructure components provided for this homework.

b. Install the application infrastructure software, and experiment with it.

c. Come up with an Enterprise application based on your own interest and the requirements set forth in the first section of this handout.

d. Install the additional sample applications, and experiment with them.

2. Prepare a short report documenting your framework-based enterprise application (using software engineering standards), and explaining its motivation.

3. Prepare a short report including functional diagrams and screenshots (as needed) to demonstrate your understanding of the infrastructure software. Explain the infrastructure software differences between the additional sample applications and the software you are using for the homework.

4. Develop and deploy your Enterprise application on top of the infrastructure software provided. Document the benefits and deficiencies of the approach on which the infrastructure software is based, and explain (as needed) how it limits your ability to develop the various application components you have envisioned for your Enterprise application. Note that you do not need to provide a complete implementation of your application in this first homework. You should restrict yourself to what you feel is feasible based on time and the level of support provided by the infrastructure software. Your application should be tuned for efficiency as allowed by the underlying infrastructure software, and you should document your performance engineering approach. You should conclude your report by suggesting, and implementing (as time allows) an improved Application Server model.

5. Extra Credit: Explain how you would build a “migration” tool that could be used to repurpose your deployed application. The migration tool should be able to extract the application details from its current source platform and redeploy it on a different target platform. Note that the target application should maintain a strict separation between content, style, and logic. Hint: Your migration tool should be able to extract and represent a generic model of your application using a suitable (markup) language.

6. Extra Credit: Implement a prototype of the analyzer tool described in question 5.

**Deliverables**

Please provide an electronic copy of your homework submission as one zip archive by sending it to the course TA by the beginning of next class. The archive should include your application software, any modified infrastructure software, a readme file describing the environment setup required to run your application, and your homework report (in word or text format). You should name your archive using the following convention for all homeworks: name_hw?_fa09.zip, where name may include additional underscore characters as needed. You are also required to provide a hard copy of your homework reports at the beginning of each session or according to the homework submission scheduled spelled out in the homework specification.

**Grading**

All assignments are graded on a maximum scale of 10 points. Your grade will be based equally on:

a. The overall quality of your documentation.
b. The understanding and appropriate use of application server related technologies.
c. Your ability to submit working and well-commented code.
d. Extra credit may be granted for solutions that are particularly creative.

Additional Information

Teaming is allowed for all the homework and projects in this course. No more than two members per team are allowed. As the course homework is based on an ongoing project, the same team members will be required work together on all of the homework assignments. Please let the TA know as soon as possible about teaming arrangements. You should only submit one report/archive per team for each assignment. To balance things out, the final grading in the course will take into account the fact that you are working as a team instead of individually, so you should feel free to work individually as well. Note that the final take home examination will require individual work.