Systems Analysis & Design
Final Exam

Jennifer Brown
7 December 2004
Dr. Ashford
# Table of Contents

## Section 1: .................................................................................................................. 2

<table>
<thead>
<tr>
<th>Question</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Feasibility:</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Technical Feasibility:</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Economic Feasibility:</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Organizational Feasibility:</td>
<td>3</td>
</tr>
<tr>
<td>B</td>
<td>Risk:</td>
<td>4</td>
</tr>
<tr>
<td>C</td>
<td>Work Plan:</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Analysis Plan:</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Work Plan</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>Outline for JAD Session:</td>
<td>7</td>
</tr>
</tbody>
</table>

## Section 2: .................................................................................................................. 10

<table>
<thead>
<tr>
<th>Question</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Packaged Software:</td>
<td>10</td>
</tr>
<tr>
<td>B</td>
<td>Server-Based Computing:</td>
<td>11</td>
</tr>
<tr>
<td>C</td>
<td>System Design:</td>
<td>12</td>
</tr>
<tr>
<td>D</td>
<td>Menus:</td>
<td>13</td>
</tr>
<tr>
<td>E</td>
<td>Normalization:</td>
<td>14</td>
</tr>
<tr>
<td>F</td>
<td>Structure Charts:</td>
<td>15</td>
</tr>
</tbody>
</table>
Section 1:

Pretend that the career services office at UR wants to develop a system that collects student resumes and makes them available to students and recruiters over the Web. Students should be able to input their resume information into a standard resume template. The information then is presented in a resume format and it also is placed in a database that can be queried using an on-line search form. You have been placed in charge of the project.

Question A--Feasibility:

How would you have assessed the feasibility of this venture when the idea first came up?

The feasibility of any system request should be measured in three areas: technical feasibility, economic feasibility and organizational feasibility. Each of these areas is covered below:

Technical Feasibility:

To assess the technical feasibility of this project one needs to look at the following factors:

• Familiarity with application
• Familiarity with technology
• Compatibility

In terms of technical feasibility, this project would be very appropriate for the University. The University already hosts both public and internal websites, and has shown a capability to design and merge data between both the Intranet sites and Internet sites. A good example is registration for classes via Bannerweb (internal), with the schedule of classes available publicly on www.richmond.edu. The system can use existing student accounts, and contact information for security. This project would be very feasible from a technology standpoint.

Economic Feasibility:

To assess the economic feasibility of this project one needs to consider the following factors:

• Development Costs
• Annual operating costs
• Annual benefits (cost savings and revenue)
• Intangible costs and benefits

Economically, this project’s feasibility is less clear than the technical feasibility. On the surface, the project does not seem to
offer a tangible economic benefit for the University. Instead, it seems that development and operating costs cannot be reclaimed by the application’s use (it’s assumed that students are able to place their resumes in the system for free).

However, the system may have intangible benefits. A resume system would increase the University’s relationship with the business community, and encouraging businesses to look at the University—possibly investing in the University’s activities.

Alumni may also see an intangible value, and the system may help keep alumni attached (and donating) to the University. The system may also increase the University’s visibility as a leading edge institution, thereby attracting new students and faculty.

But these benefits are very tangential to the system, and while the University may have the economic resources to develop this project, it may not show any economic benefit to the University in a direct relational way.

**Organizational Feasibility:**

To assess the organizational feasibility of this project one should consider the following:

- Project champions
- Senior management’s support
- Users
- Other stakeholders
- Is the project aligned with the business?

The organizational feasibility of this project is not altogether clear. The project has sponsorship by the career services office, and it is assumed the department has budgeted the necessary funds to the IS department to manage the project.

While the project may have sponsorship and management, the biggest issue is whether or not both students and businesses will use it. Students may use other resume services such as Monster.com, and unless there is a benefit to posting to the University’s site, they may continue to use these other services. Either the resume entry must be an easy task, or businesses presence must be high enough to make posting the resume worthwhile to students.
Businesses must also perceive a benefit from the system. The system should provide accurate results for searches. Businesses must be “sold” on using the system to attract candidates. The marketing of the system will be very important going forward. Otherwise, it will be a system that is not used, and that is not a feasible system.

**Question B--Risk:**

*How risky would you have considered this project, and why?*

On the surface, this project does not seem very risky for the University. The technical abilities of the IS department, and the existing infrastructure mean that developing and maintaining this system would be very simple for the University. The University has the economic resources to support this project as well.

The real risk for this system lies in whether or not the system will be used. Based on the description above, it is uncertain whether students or business have expressed a need or desire for the system. Further analysis with the potential users of the system should be performed to ensure the system will be used.

In terms of impact to the University’s current infrastructure, the system would not be a mission critical application—the University would not shut down if the application went offline for any reason. The system would likely experience heavier loads during the Spring semesters, and during the Summer, when recent graduates are looking for jobs, and this load would have to be balanced against the fact that the entire University system is busier during certain parts of the year.

In all, the project is not very risky—with the exception of ensuring that students and businesses would use the application.
Question C—Work Plan:

Develop a work plan that lists the tasks that will need to be completed to meet the project’s objectives. In the development of this plan, include an analysis plan for this project (items to consider/evaluate), develop an outline to be used in a JAD with regards to goals to be achieved in gathering information about this project.

Analysis Plan:

The following items need to be researched and addressed before beginning this project, these items should also be addressed within the first few JAD sessions of the project:

- **Organizational Feasibility:** This project will require further analysis regarding whether or not students and business will use the final application. Before any work begins on this project, it should be determined whether the potential users of the application are interested in using the application. If the final application will be useful to both the students who enter their resumes, and the businesses that will query the system for possible candidates, development should go forward.

- **Lifecycle Planning:** Another area that needs to be analyzed further is the lifecycle of the data within this project. For instance:
  - How long will a student’s resume be active within the system?
  - When a student is hired, how will their resume information be handled?
  - How and when will students be able to update or change information within the system?
  - How and when will businesses be able to update the system?

- **Defining User and Access Requirements:** The potential users for this system are only vaguely described in the above. Further research will be required to determine:
  - What levels of security are required for the system? Should students be able to view other student’s resumes, or should only businesses be allowed to view resumes?
  - What kinds of information need to be captured for student resumes?
o What kinds of information should be captured from businesses? Should businesses be able to post job listings, or only search the system for possible candidates?

o How long can a student keep a resume on the system?

o Can alumni use the system, or should it only be restricted to current students?

• **Integration Issues:** The project team also needs to further analyze issues surrounding integrating the system with the current University web architecture. This includes determining the hardware that will host the system, and the ways in which students and businesses can access the system. The analysis should also consider issues that might arise if the University switches to a new type of web-server, or sets different requirements for database or user interfaces.
Work Plan

The following is a rough Gantt Chart from Microsoft Project outlining the work plan for the project:

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Duration</th>
<th>Start</th>
<th>Finish</th>
<th>Predecessors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Begin Project Planning</td>
<td>0 days</td>
<td>Tue 12/7/04</td>
<td>Tue 12/7/04</td>
<td></td>
</tr>
<tr>
<td>Create Formal System Request</td>
<td>3 days</td>
<td>Tue 12/7/04</td>
<td>Thu 12/9/04</td>
<td></td>
</tr>
<tr>
<td>Create Feasibility Analysis</td>
<td>2 wks</td>
<td>Thu 12/9/04</td>
<td>Wed 12/15/04</td>
<td></td>
</tr>
<tr>
<td>Develop Risk Assessment</td>
<td>1 wk</td>
<td>Mon 12/13/04</td>
<td>Fri 12/17/04</td>
<td></td>
</tr>
<tr>
<td>Formalize Project Plan</td>
<td>3 days</td>
<td>Thu 12/19/04</td>
<td>Mon 12/27/04</td>
<td>2,3,4</td>
</tr>
<tr>
<td>End Project Planning</td>
<td>0 days</td>
<td>Mon 12/27/04</td>
<td>Mon 12/27/04</td>
<td>5</td>
</tr>
<tr>
<td>Project Analysis</td>
<td>0 days</td>
<td>Mon 12/27/04</td>
<td>Mon 12/27/04</td>
<td>5</td>
</tr>
<tr>
<td>Develop Formal System Proposal</td>
<td>1 wk</td>
<td>Tue 12/28/04</td>
<td>Mon 1/3/05</td>
<td>5</td>
</tr>
<tr>
<td>Develop Requirements Definitions</td>
<td>2 wks</td>
<td>Tue 1/3/05</td>
<td>Mon 1/3/05</td>
<td>8</td>
</tr>
<tr>
<td>Develop Use Cases</td>
<td>2 wks</td>
<td>Tue 1/3/05</td>
<td>Mon 1/3/05</td>
<td>9</td>
</tr>
<tr>
<td>Develop DFDs</td>
<td>2 wks</td>
<td>Tue 1/3/05</td>
<td>Mon 1/3/05</td>
<td>9</td>
</tr>
<tr>
<td>Develop ERDs</td>
<td>2 wks</td>
<td>Tue 1/3/05</td>
<td>Mon 1/3/05</td>
<td>9</td>
</tr>
<tr>
<td>End Project Analysis</td>
<td>0 days</td>
<td>Mon 1/3/05</td>
<td>Mon 1/3/05</td>
<td>12,11,10</td>
</tr>
<tr>
<td>Begin Project Design</td>
<td>0 days</td>
<td>Mon 1/3/05</td>
<td>Mon 1/3/05</td>
<td>13</td>
</tr>
<tr>
<td>Set up physical hardware and software for</td>
<td>1 wk</td>
<td>Tue 1/3/05</td>
<td>Mon 1/3/05</td>
<td>9</td>
</tr>
<tr>
<td>Develop Database</td>
<td>4 wks</td>
<td>Tue 1/3/05</td>
<td>Mon 2/21/05</td>
<td>15</td>
</tr>
<tr>
<td>Design User Interface</td>
<td>7 wks</td>
<td>Tue 1/3/05</td>
<td>Mon 3/1/05</td>
<td>15</td>
</tr>
<tr>
<td>End Project Design</td>
<td>0 days</td>
<td>Mon 3/1/05</td>
<td>Mon 3/1/05</td>
<td>17,16</td>
</tr>
<tr>
<td>Begin Project Implementation</td>
<td>0 days</td>
<td>Mon 3/1/05</td>
<td>Mon 3/1/05</td>
<td>18</td>
</tr>
<tr>
<td>User Testing</td>
<td>1 wk</td>
<td>Tue 3/1/05</td>
<td>Mon 3/2/05</td>
<td>19</td>
</tr>
<tr>
<td>Install System</td>
<td>2 days</td>
<td>Tue 3/2/05</td>
<td>Wed 3/23/05</td>
<td>20</td>
</tr>
<tr>
<td>Implementation Changes</td>
<td>1 wk</td>
<td>Tue 3/15/05</td>
<td>Mon 3/21/05</td>
<td>19</td>
</tr>
<tr>
<td>End Project</td>
<td>0 days</td>
<td>Wed 3/23/05</td>
<td>Wed 3/23/05</td>
<td>21,22,20</td>
</tr>
</tbody>
</table>
Outline for JAD Session:

The following could be used as a rough outline for the JAD sessions that would determine the requirements for this system. Each bullet point might be the topic for a single session.

I. What is the potential value of this system?
   a. Value to the University
   b. Value to Students
   c. Value to Area Businesses

II. What are some of the costs that could be incurred with this system?
   a. Physical costs (hardware/software)
   b. Personnel costs (support/developers)
   c. Maintenance costs

III. What are some of the risks that could be associated with this project?
   a. Systems risks
   b. Usability risks
   c. User risks

IV. Who are the potential users for this application?
   a. University users
      i. What purpose would the application serve them?
   b. Student users
      i. What purpose would the application serve them?
   c. Business users
      i. What purpose would the application serve them?

V. What are the operational requirements for this system?
   a. Access requirements (user levels?)
   b. Data storage
   c. How can this application be integrated in with the existing University Web architecture?
   d. How will this application mesh with the business partner’s systems?

VI. What are the performance requirements for the system?
   a. What will be the average load on the system?
   b. What would be the peak load?
   c. What are the up-time requirements for the system?
   d. How secure does the information within the system need to be?

VII. What are the functional requirements for the system?
   a. What kinds of information will need to be inputted?
      i. By Business contacts
      ii. By students
      iii. By the University
b. What will the final output information be?
   i. For Students
   ii. For the University
   iii. For Businesses?

c. Are there additional add-ons (spell-checking/e-mail and advisor etc) that need to be considered?

VIII. How will the user flow through the application?
   i. If they are a business partner?
   ii. If they are a student?
   iii. If they are part of the University?
Section 2:

**Question A—Packaged Software:**
What are some problems with using a packaged software approach to building a new system? How can these problems be addressed?

Using packaged software can either be a quick and easy solution, or a long and tiring nightmare for organizations. While packaged software seems a quick solution to complex problems, sometimes the software is not the ideal fit for the organization.

Organizations that decide to use packaged software may run into several common problems. The first is the ability of the packaged software to match the basic requirements of the project. For instance, a store might want to create a system that tracks inventory, and prints pricing stickers for received product. If the company selects a packaged software product, it may limit some of the functionality of the new system. The company may currently use pricing labels in a specific format: with the product name, the store logo, and the price printed on it, but the packaged software may only print a pricing sticker with a UPC label, and a price on it.

Companies can overcome these limitations by thoroughly researching available packages before deciding on one, or working with the software company to make some moderate customizations to the package. The other solution is to change the processes the company uses. In this example, if the company really likes the way in which the inventory system works, they may be willing to sacrifice the pricing sticker requirement for the other benefits of the packaged software.

Another issue that arises with packaged software is integrating the software with legacy systems. For the inventory system listed above, integrating the data in the current inventory system with the new system may prove arduous, or nearly impossible depending on how the legacy system is designed. This may require a programmed work-around to get the information into the new system. Another issue with this system may arise in the point-of-sale system. If the system is
not designed to recognize the data format used by the inventory system, the new system will be useless, and will not meet the requirements of the project.

One way to tackle integration issues is to reengineer the legacy system, or seek a package that is compatible with the existing system. The point-of-sale system could be redeveloped to match the data in the new inventory package, or the store could select a package that states that it is compatible with the point-of-sale system. Either way, the solution would require the point-of-sale system to be included in the project, along with the original inventory system.

A final issue with packaged systems arises around finding support for the purchased system. If a package is adopted and customized, any changes in the system may result in problems with the application that may or may not be supported by the software’s vendor.

A good example is operating systems. Almost all organizations run on purchased operating systems. When Microsoft stated that it would no longer support Windows 2000 after 2004, many companies were forced to upgrade to Windows XP Professional. This upgrade affected many existing systems, and programs. Many companies had to dig through their entire software portfolio to determine what would, and would not work with Windows XP. In some cases all that was needed was an upgrade, in other cases entirely new packages had to be purchased, because the vendor no longer produced the product, or was no longer in business, and as a last resort, some packages had to be completely redeveloped.

**Question B—Server-Based Computing:**
What is the biggest problem with server-based computing?

Server-based architecture is the oldest systems architecture, and has several problems. Most of which arise from the fact that the server is the center of the entire system.

The first issue with a server-based system is that the server does all the processing, all the data storage, and all the work for the system. As users, data, and processing needs increase,
increasing demand is put on the server to handle all aspects of the system. This increase will slow the system down, and eventually require expensive upgrades.

Another issue with server-based computing relates to the above. Because a server is the entirety of the system, a failure in any component can be catastrophic to the entire system. A power outage, or even buggy code can render the data, interface, and communications completely useless until the issue is repaired. In other designs, certain components may be lost in a catastrophic failure, but the system can be used in a crippled state while changes are being made. With a server-based architecture, the entirety of the system is exposed to this risk.

**Question C—System Design:**

How can a system be designed to be used by both experienced and first-time users?

There are several techniques that can be used to create a system that is both usable for experienced and first-time users. The same basic rules of good design apply regardless of the level of experience the users have.

The first technique is to ensure design is consistent across the entire application. For instance, if names or titles appear across the top of a page, it should stay that way throughout the application. The section name should not jump from the top of the screen, and then to the bottom in another section. Likewise, the organization of information within the application should remain consistent. If areas with additional information are hidden with down arrows in one section, all areas with hidden information should use down arrows. If red text indicates an error in the data, it should not also be used to indicate necessary information that needs to be entered on that page. The way that information is entered into the application should be consistent, also. If one section uses a pull-down menu to select a customer’s gender, another section should not use radio buttons for the same selection. If field labels appear above the fields in one section, they should not be next to the field in another.

A second technique is to create a built-in help system for novice users. Tool-tips—those little boxes of information that pop
up when you mouse over an icon—are an example of built-in help for new users. Another is a “Help” menu option. If a novice user is stuck, they can click on a help icon, or select “Help” from a menu, and receive step-by-step instruction on what they need help with. Many well-written applications will include the user guide within the built-in help system for the application.

An emerging solution helping design address the needs of both novice and experienced users is the ability for the user to customize the interface. While new users prefer to use clickable icons, and all options available via a menu, experienced users tend to prefer contextual menus, and keyboard shortcuts. Good modern interfaces will initially be set up for the novice user—with all options in the menu, large icons, explicit directions on screen, and things like tool-tips triggered when someone hovers on an icon for a specified period of time. When a user has become familiar and comfortable with the system, he or she can select a customize option, and turn on or off the options he or she wants in the application.

**Question D--Menus:**

**Why are menus the most commonly used navigation control?**

The simplest explanation for why menus are used for navigation control is that they are the de facto standard for navigation in a GUI, and they are what users are expecting. They present available options on screen for the user, rather than requiring the user to memorize commands. Likewise they allow like tasks to be grouped together for access. A user should be able to click on the “Edit” menu and see the options that are related to editing—such as “copy”, “paste”, “cut” and “select all.” Since menus are fairly universal in today’s applications, most users know what to expect underneath the common headings.

A major advantage of menus over other navigation controls is that a user can surf menu options. With typed commands, the user is committed to the command once it is entered, likewise with direct manipulation, the change may be made before the user realizes what a particular change is. With a menu, a user can click on the headings, view the options, and find the right one to meet his or her needs before committing to that option.
**Question E--Normalization:**

**What is the purpose of normalization?**

Normalization serves several purposes. Firstly, it helps eliminate wasted space. Using the in-class example, breaking out each item on a customer’s order into its own table saves space in instances where a customer only orders one item. In addition, space would exist if a customer ordered more than three items. Neither of these options were available in the initial table.

A second purpose of normalization is to eliminate possible conflicts of information in the system. In a large database, customer information might be needed for the marketing mailing lists, by the billing department for credit processing, and by the order-processing department to ship the product to the customer. If each department had customer’s information its own table, it’s possible that if a customer moves, the data in one department’s table might be updated, but it would not be updated across the system. If each department’s table references a separate customer table, when a customer moves, that information is updated for all departments—eliminating the possibility of conflicts with this information.

Finally, normalization helps to eliminate redundancy in the database. This can be explained by using the example of calculating the shipping cost for a catalog order based on the customer’s preference. If only one table was used for an entire order, if 20 customers chose to use regular post, and 10 chose FedEx there would be 20 entries for regular post with the associated cost, and 10 for FedEx with that cost. With normalization, a shipping table would hold the information about the cost of shipping by post, and only need a second records to hold the information about FedEx. The order table only needs to reference the appropriate record in the shipping table to calculate the postage for the customer.
**Question F—Structure Charts:**

*Why is it desirable for a structure chart to be highly cohesive and loosely coupled?*

Good structure charts, and well-designed systems are highly cohesive and loosely coupled. Cohesion in modules refers to how many functions a particular module performs. A module that only performs one function—for instance a module that calculates the total price of goods sold would be a highly cohesive module. The reason high cohesion is desirable is because it means that the module is more portable, and adaptable as the application develops. A module that calculates the total price can be applied to a cash register sale in a physical store, or a catalog, or Internet sale. Whereas, a low cohesion module—like one that calculates the total sale and the sales tax could not be used outside of the physical store.

Coupling refers to how independent each module is from the others. Modules that are able to perform a function without relying on other modules are loosely coupled. Loose coupling is desirable because it prevents issues with a change in one module cascading through the application, and causing problems. Using the in-class example, if a module calculates the date with a two-digit year, if other modules rely on that function, and the date format is changed to a four-digit year, it could affect the ability for the other functions to calculate. For instance 1998 in the first module would equal January 9, 1998, but after the change would equal the year 1998. Ideally, only the necessary information should pass between modules, and it should be passed at a minimum.