As discussed in class, we will re-do the Deco_Bike_Share information system DFD. I offer the following points to assist you:

- The problem requires you to create both a Context and a Level_0 diagram.

- It would be helpful for you to visit the website and view the little videos to “get” how the system works.

- The actual system supports choice of language (an input from User to System); let’s NOT worry about implementing Language-choice; extra-credit for those who successfully incorporate language choice.

- The inputs and outputs on the Context diagram (and also required on the Level_0 diagram to maintain “balance”) will include

<table>
<thead>
<tr>
<th>Inputs from User</th>
<th>Outputs to User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Membership Card data</td>
<td>Map showing bike-share locations</td>
</tr>
<tr>
<td>Non-Member Credit-Card data</td>
<td>Current Count of bikes at a bike-share location</td>
</tr>
<tr>
<td>Chosen Bike ID number</td>
<td>Notification of failed membership card</td>
</tr>
<tr>
<td>Request for count of available bikes at a specific Bike-share station</td>
<td>Credit-card receipt (for non-member)</td>
</tr>
<tr>
<td>•</td>
<td>• Credit card rejected message</td>
</tr>
<tr>
<td>•</td>
<td>• Audible ‘bike-check-in’ sound</td>
</tr>
<tr>
<td>• Perhaps others</td>
<td>• Perhaps others</td>
</tr>
</tbody>
</table>

- I believe the only other needed sink/source is Credit-Card-Company; Output to CCC will include Non-Member-Credit-Card-data and rental fee. Input from CCC includes credit-card-charge successfully processed, and credit-card-charge failed (for whatever reason).

- Google Mapping is not required. The Map showing bike-share locations is static, not requiring any real-time rendering (as was the case with ALT fueling station system). This static map can come from an internal data-store.

I suggest the following sub-processes on the Level-0 diagram:

- Verify membership
• Accept payment from non-member
• Check-out a bike
• Check-in a bike
• Perhaps others

A data-store will be required to track the real-time count of bikes on hand at each station (like the inventory system, with increments and decrements)

A data-store will be required to track bike-check-out and bike-check-in times, allowing for billing for over-time use of the bike (user is supposed to return bike within 30 minutes; if late there are added fees)
Decobike Bike Rental Level-0 Diagram

1.0 Format Bike-Share Location Map
- Request for Bike-Share Location Map
- Formatted Bike-Share Location Map

2.0 Calculate Available Bikes at User-Selected Location
- Query Bike-Share Location Map
- Bike Inventory Database
- Bikes, Availability, and Locations Raw Data

3.0 Verify Membership
- Membership Card Data
- Notice of Failed Membership Card
- User Membership Status

4.0 Process Payment from Non-Member
- Non-Member Credit Card Data with Rental Fee
- Credit Card Successfully Processed Message
- Credit Card Rejected Message

5.0 Check out a Bike
- Payment Confirmation
- Bike ID and Check-In Time
- Check-in and Check-Out Times

6.0 Check in a Bike
- Bike ID and Check-In Time
- Bike Availability and Location Data for User Returned Bike

User

Credit Card Company
Part 2:
Create a Level 0 DFD for this version of the ParkIT_DC Context Diagram:

What is Park It DC?
Park It DC is an application that allows you to check a specific area in the district for parking information. Why not check parking information before you leave and see what streets you can park on. Check what meters cost money and what ones are free. Park It DC will even help you not get a ticket for parking in an RPP zone!

How does it Work?
In Short, we store all data in a database. Every 10 mins. we make a call to the crime and service request feed and either update the already existing record or insert a new one. Then we use Google Maps API calls to read XML files that we generate from the database and plot the data on the map.

RPP = Residential Parking Permit (free parking for 2 hours for non-registered cars, unlimited parking for registered cars)

Unfortunately, the real ParkIT_DC system is no longer online. The system DID support input of a neighborhood in DC, and would provide output of various data (parking lots, open parking meters, RPP areas, and current crime activity) back to the user, all rendered on a GIS map.

Data Flow Diagram (context level) for Park It information system:
MEMORANDUM

TO: DIRECTOR OF INFORMATION SYSTEMS
FROM: CAITLIN HOMENDA
SUBJECT: OPEN SOURCE PLATFORMS
DATE: SEPTEMBER 14, 2014
CC: PROFESSOR TERRY USREY

Per your recent request, I have compiled information regarding the benefits and challenges associated with adopting an Open Source platform.

Benefits of Open Source Platforms
There are several benefits associated with adopting an Open Source platform. Firstly, since the source code is open and accessible to any user, the source code is being regularly and continuously analyzed by a large community. This means that if there are issues with bugs or errors in the code, they are often found and fixed more expeditiously than for proprietary software since a larger number of individuals are reviewing the code. Secondly, having access to the source code for the programs our department uses allows us to innovate and customize the software for our specific purposes. Since proprietors are often attempting to develop products that fit a wide range of clients, proprietary software does not always best fit our specific needs. Using an Open Source platform would allow our developers to make customizations, as needed, to fit our department’s changing and unique needs. Thirdly, Open Source software is not dependent on one proprietary company. For proprietary software, if the proprietor company dissolves, the software may no longer be upgraded or updated, leaving our department to need to migrate to new software to meet whatever needs that specific piece of software met. This same situation would be less likely to occur with Open Source software as many individuals are likely to continue to use, develop, and update the software even if the originating organization no longer has any influence on the development of the software. Finally, migrating to an Open Source platform would increase the interoperability of our current systems. Open Source software is often developed using the same open standards, so these different software programs would work well together and allow our department to standardize all our systems. This is not always the case with proprietary software – it does not always function well with other programs or operating systems other than those owned by the software’s proprietary company.
Challenges
While there are several benefits to adopting an Open Source platform, there are some challenges associated with this migration. Firstly, the migration could be fairly costly if our internal developers needed extensive training in order to understand the different software associated with the adopted Open Source platform. Additionally, it may be difficult to transfer all of our data from our current disparate systems to the Open Source platform. Secondly, if our department decided to customize the software to meet our specific needs, we would either need our developers to be very well-trained to do so, or depend on an outside individual to customize the systems for us. If we are dependent on a third party to customize the software for us, we would be in a similar position as if we remained with proprietary software. Should the individual decide to no longer assist us with our customized software, we would either need to switch to a less customized version or train an in-house developer to be able to maintain the customized software. Finally, there are some security concerns with Open Source software. Since the source code is viewable by anyone, it may be more susceptible to malware and other viruses.

Conclusion
While there are several challenges associated with adopting an Open Source platform, such as the initial cost, training needed, and security concerns, the benefits of customization, lack of dependence on a proprietor, and interoperability of Open Source software make this a worthwhile endeavor for our organization.

Part 4:
Provide brief answers to Review Questions 2,5,7,13 ClassPak page 48, and Discussion Question 3 Class Pak page 49.

Review Questions

2.) Describe the key activities performed by IS professionals in each step of the SDLC.
Definition Phase
   Feasibility Analysis
      In the Feasibility Analysis portion of SDLC, IS professionals assess the propose system’s technical feasibility and assist in the preparation of a cost-benefit analysis of the proposed system based on the anticipated development costs and estimated developer time needed.
   Requirements Definition
      In the Requirements Definition portion of SDLC, IS professionals confirm that developers have sufficient definitions from the users in order to write the code and to make sure the system will work in the user-intended manner. They also assist in drafting and revising the requirements specifications document.
Construction Phase

System Design
In the System Design portion of SDLC, IS professionals design the physical system, including the software/hardware that will be used and the structure of databases. The system design stage is based on the prior-developed conceptual requirements document.

System Building
In the System Building portion of SDLC, IS professionals build the computer programs and develop the databases that will be used by the system based on the design document from the prior stage of the process.

System Testing
In the System Testing portion of SDLC, IS professionals complete initial testing on the system to confirm that all parts work appropriately. Additionally, they correct any bugs they or other users testing the system find.

Implementation Phase

Installation
In the Installation portion of SDLC, IS professionals convert any relevant data from the prior systems to the new system. They also install any necessary hardware or software that is required to operate the system.

Operations
In the Operations portion of SDLC, IS professionals operate the new system in production mode along with other users of the system. They also provide support for other personnel using the system.

Maintenance
In the Maintenance portion of SDLC, IS professionals make any necessary changes to the system after it has been put into production. They may adapt the system to meet changing needs, add enhancements, or fix bugs.

5.) Describe a distinct advantage of each of the four strategies for implementing a new system, as shown in Figure 10.4.

Parallel Strategy:
In this strategy, a department runs both new and old systems at the same time during a short transition period. This strategy may make it easier for end-users to find issues since they are comparing the data found in both systems. Additionally, this method gives users time to adapt and feel confident with the new system before moving fully to it.

Pilot Strategy:
Since this strategy implements the new system in only one part of the organization, it allows time for bugs and other issues to be figured out and resolved prior to the whole organization using it. If major problems are discovered in the new system, it only slows down the area participating in the pilot rather than the whole organization.
Phasing Strategy:
In this strategy, parts of the new system are phased in over time. This allows the firm an opportunity to benefit from the new system more quickly than other strategies, but also allows a large system to be broken down into smaller projects. Having smaller projects allows IS professionals to focus their attention to issues related to the implementation of each phase. Additionally, it might reduce anxiety that end users have with changing systems as they will have an opportunity to slowly adjust to the new system.

Cutover Strategy:
In the cutover strategy, an organization moves to a new system on a specific date with no further reliance on the old system. This method is the fastest implementation strategy and requires that end-users adjust more quickly to the new system as they do not have the crutch of old system availability.

7.) Why is an accurate and complete requirements definition especially critical when using the SDLC “waterfall” approach?
If the system requirements are not defined clearly up front, it could cost the organization sufficient funds as the developers may misinterpret what end-users need from the system and develop a system that is not useful or efficient for end users. Fixing these issues after the system has already been implemented is much more time-consuming and costly than clearly defining any system requirements and user needs at the beginning of the process.

13.) Why does the use of contractors increase the complexity of an IT project?
Added complexity arises from the organization not having direct oversight on the contracted workers. Additionally, if the contractors are off-shore, there may be added complexities from time zone differences, language/cultural barriers, and concerns with respect to the privacy of intellectual property.

Discussion Question
3.) There have been many failures in the development of application systems using the traditional SDLC. Discuss some characteristics of the methodology that could contribute to the high failure rate under certain situations.
One difficulty associated with the SDLC process is that it is a long-term, costly project. Some businesses may have started using this approach before realizing the length of time and funds needed to utilize the SDLC process. The operation may have eventually failed due to an attempt to cut costs by cutting corners at some stage in the process (either monetarily or time-wise). Additionally, since this is a long-term project, it can be difficult to adapt it to changing needs and requirements of the office. This could mean that the project is outdated by the time it can be fully implemented. Additionally, if the definitions were not clearly defined in the Definitions Phase, it could have led to increased costs and difficulty correcting these issues later in the project. This may have cause organizations to abandon the project rather than fix the issues at such a late stage in the project. The complexity and length of investment in the process are likely the leading contributions to the high failure rate.