TIM 50 - Business Information Systems
Lecture 14
Instructor: John Musacchio
UC Santa Cruz
May 15, 2014

Class Announcements
- Database Assignment 2 posted
- Due 5/27
- Business paper draft due 5/20

Aside: Network Effects
- The value of owning some products goes up if lots of other people have it too.
  - Examples?
- This phenomenon is called “network effects”
- How do standards influence network effects?

Network effects
- Standards can harness network effects to the industry advantage
  - Revenue = (market size) x (market share)
  - Increases value to customer
  - Increases competition
    - Only within confines of the standard
    - But forces customer integration or services of a system integrator

Why standards?
- *de jure* are customer driven to reduce confusion and cost
- *de facto* standards are sometimes the result of positive feedback in network effects
- Customers and suppliers like them because they
  - increase value
  - reduce lockin
- Governments like them because they
  - promote competition in some circumstances
  - May believe they can be used to national advantage

Approaches
- Consensus
  - ISO
- Collaborative design
  - MPEG
- Competitive “bake off”
  - IETF
Open vs. Proprietary Standards

- Open standard – a standard that is well documented, unencumbered by intellectual property rights and restrictions, and available to any vendor.

- What are the advantages?

- What are the disadvantages?

Why companies participate

- Pool expertise in collaborative design  
  - e.g. MPEG

- Have influence on the standard

- Get technology into the standard
  - Proprietary, with expectation of royalties
  - Non-proprietary

- Reduced time to market

Standards applied to Business Processes?

- Can you standardize business processes?

- Yes!
  - ISO 9000
    - A set of standardized business processes for Quality Management.
    - Supports TQM (Total Quality Management)
  - RosettaNet
    - A set of standardized business processes, and accompanying standardized data interfaces/formats for conducting e-business.

Chapter 9

Applications and the Organization

Build vs. Buy?

Purchase off the Shelf
  - less time and cost
  - benefits of using a “standard” solution
  - support available
  - must mold org to app
  - no potential for competitive advantage

Outsource
  - developers not as familiar with org as you
  - more opportunity for customizing than off the shelf
  - contractor may share knowledge with competitors
  - contractor may have too much bargaining power

Make
  - most customizable of 3
  - easier iteration between conceptualization and development needed
  - most risky
  - org may lack competency to do it

Application Lifecycle

- It is important to think beyond acquiring an application
  - How do we come with the idea?
  - How do we architect it?
  - How do we implement?
  - How do we extend and maintain it?

- For this reason, the software engineering community came up with:
  - Application Lifecycle Model
Application Lifecycle

Stages:
1. Conceptualization
2. Analysis
3. Architecture Design
4. Development Evolution
5. Testing and Evaluation
6. Deployment
7. Operations, Maintenance, and Upgrade

1) Conceptualization

- What is the vision?
  - What are the objectives?
  - What is the business case?
  - EXAMPLE: Seatback system to sell seat swaps
- Business Case:
  - Increase revenue, passenger satisfaction

Conceptualization

- New in-flight seatback system
  - Sell upgrades and seat swaps
  - (People who want to get away from sick people ...)
  - Offer to exchange seats

2) Analysis

- Describe what the application will do.
- Enough info to allow "stakeholders" to review idea
- Don’t make highly detailed specifications
- Describe scenarios in which it is used
  - (Use Cases)

2) Analysis -- Example

- Example: Scenario:
  - Seat Trade
    - Passenger in 10C (aisle) offers to trade seat for frequent flyer miles
    - Business traveller in 20B (middle) offers to pay $500 to get aisle seat

3) Architecture Design

- Decompose the application into subsystems
  - Hardware, software
  - Try use commercial off the shelf subsystems
  - Try to use standard infrastructure layers
  - Operating system, network, middleware, etc.
Architecture

When a module is composed of sub-modules, the architecture is hierarchical.

3) Architecture Continued

- Define the functionality, interaction and interfaces of subsystems
- While doing this, consider
  - Scalability
    - How easily can we increase the number of users and maintain performance?
  - Extensibility
    - How easily can we add new features in the future?
  - Administration
    - How much work will it take by humans to keep this running properly?

4) Development Evolution

- Develop the details
  - Develop/program custom subsystems
  - Have contractor build outsourced pieces
  - Put together with off-the-shelf components
  - Incremental
    - Start with simplest implementation and get it working
    - Later add more features.

5) Testing

- A must!
- If architected well, we can test subsystems independently.
- Alpha test – offline test of prototype
- Beta test – test in intended environment with cooperative users
  - Example – give HHC to initial group of FA’s

6) Deployment

- Convert from previous processes if necessary
  - Example: CISCO ERP (all at once)
  - Or, you could do incrementally
- Train users
- Data importation
  - (if necessary)
7) Operations, Maintenance, Upgrade
- Maintain Security
- Repair Problems
- Correct performance short comings (Cisco ERP)
- Add features

Application Lifecycle Model
concluding remarks
- ALM rarely followed precisely
- Many times projects loop between stages
- ALM followed more closely in larger companies
- Alternative:
  - Rapid Iterative Prototyping
    - (Cisco did some of this in the ERP case.)