Class announcements

- Final Exam:
  - Wednesday 12/10, 12-3pm, Kresge 321
Student Presentations

- Jiabi(Stephy) He
- Jeffrey Hogg
Akamai Case
Akamai Freeflow

Local Office or ISP

Web Page

Text...

Large Company

Web Server

INTERNET

NSP 1

NSP 2
Freeflow

- Deployed in 1999
- Akamai Infrastructure
  - 13000 servers in 954 networks by 2001
- Customers -
  - Large Commercial Websites
- Revenue model - $2000 per mbps served
  - (For comparison, normal Internet access cost 500 mbps at time)
2000 Financials

- $196 Million Loss (Before special charges)
- $90 million revenue
- %20 gross margin, after deducting
  - server depreciation
  - payments to network partners
  - Data center space
- But, most expenses of shouldn’t grow at same rate as number of customers, so margin should improve

- $201.5 million SG&A
  - (selling general and administrative)
  - (largely sales force cost)
  - Again, this might not grow at same rate as the number of customers.
- $40 million R&D
Competition

- Hosting firms (substitute)
  - Exodus
- Other CDNs
  - Sandpiper, Adero, Mirror Image
- Content Alliances
  - Akamai’s competitors banded together to share networks
2001 Market Changes

Bad
- Dot-coms bust
- Customers leave
  - “churn rate goes to 22% per quarter”

Good
- Hosting firms go bust (exodus)
- Some CDN competitors go bust.
- Competing CDN alliances mired in problems
Akamai EdgeSuite

Local Office or ISP

Web Page

Movie A

Movie C

Construct Page

Movie A

Movie A

Movie C

Movie C

Movie C

Movie B

NSP 1

NSP 2

INETET

Large Company

Web Server

Movies in Santa Cruz after 8pm?
EdgeSuite

- Assemble dynamic pages at edges rather than just serve heavy objects
- Value proposition
  - Performance improvement
  - Cost and complexity reduction
  - Scalability
  - Security
- Pricing - higher than old service
- Soon edge suite dominated revenue
Technology

Dynamic CDN technology: ESI (edge sides includes)

Develop as open standard why?

Akamai not big and credible enough to force a de-facto standard on market
Marketing

- **Difference in selling old vs new products:**
  - **Old product**
    - Geared toward speeding up websites
    - Revenues of their clients depended on speed
    - Easier to get sale
  - **New Product**
    - Simplify company IT function
    - Cost vs. revenue center
    - Harder sell. More data driven...
  - Consequently new product needs more professional sales force

- **Channels?**
  - Distribution Partners (IBM) credibility
  - Direct Sales Force too
# Recent Performance

## Consolidated Statements of Operations Data:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Revenues</strong></td>
<td>$210,015</td>
<td>$161,259</td>
<td>$144,976</td>
<td>$163,214</td>
<td>$89,766</td>
</tr>
<tr>
<td><strong>Total cost and operating expenses</strong></td>
<td>$101,948</td>
<td>$12,370</td>
<td>$52,780</td>
<td>$2,171,108</td>
<td>$989,359</td>
</tr>
<tr>
<td><strong>Net income (loss)</strong></td>
<td>$34,364</td>
<td>$(29,281)</td>
<td>$(204,437)</td>
<td>$(2,435,512)</td>
<td>$(885,785)</td>
</tr>
<tr>
<td><strong>Net income (loss) attributable to common stockholders</strong></td>
<td>$34,364</td>
<td>$(29,281)</td>
<td>$(204,437)</td>
<td>$(2,435,512)</td>
<td>$(885,785)</td>
</tr>
</tbody>
</table>

(In thousands, except share data)
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: HHC to inform flight attendants which passengers are low and high value.

- Business Case:
  - Increase repeat business from high value customers.
1) Conceptualization -- Example:
2) Analysis

- Describe what the application will do.
- Don’t make highly detailed specifications
- Describe scenarios in which it is used
  - (Use Cases)
Example: Scenario:

- Premium Passenger Identification function

- FA wants to see who are premium passengers to offer them a free drink

- FA clicks “on” premium passenger radio button

- HHC displays color coded seat map.
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

HEADQUARTERS

Airline Dataserver

Airline Intranet

HHC Server

Wireless Link

HHC

Airline
3) Architecture Continued

- Define the functionality, interaction and interfaces of subsystems

- While doing this, consider
  - Scalability
    - Can we increase number of users easily?
  - Extensibility
    - Ability to add new features later
  - Administration
    - Is it hard to keep it working?
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
6) Deployment

- Convert from previous processes if necessary
  - Example: CISCO ERP (all at once)
  - Or, you could do incrementally
- Train users
  - Example: Frito-Lay HHC
- 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.)