Derivation and Evaluation of Business Blueprints

Zi Zhou Wang (zw3948)

EE 382C Software Architectures
Dr. Barber

March 23, 2018

Part 1: Deriving Deployment Blueprints (DB)

Deployment Blueprint #1 (DB1) – Automated Solution

Satisfaction of domain functions, data and I/O by solutions

SB Solution Component	BB Functions Satisfied	BB Data Satisfied	I/O Satisfied
Item Browsing	- Select Item in Store - Add Item to Cart - Make a Purchase - Reorder Subscription - Create Subscription - Add Item to Wishlist - View popular Products - View Similar Products - Compare Items - Edit or Delete Subscriptions	- Item Selected - Shopping Cart - Confirmation /Order Numbers - Items successfully purchased - Subscription orders - Order - User Subscriptions - Order History - List of Orders the user has placed - Items from Past Order - User wishlist	- Verify Item Available in Inventory requires Item from Add Item to Cart - Check Expiration Date requires Selected Item from Select Item in Store - Add Item to Wishlist requires Selected Item from Select Item in Store - Compare Items required Items which is retrieved directly from Shopping where it is allocated - View Popular Products requires Item Inventory retrieved directly from Inventory where is is allocated - View Popular Products requires Item View Counts retrieved directly from Inventory where is is allocated
Payment	- Enter Payment Information - Send Notifications - Schedule Refund - Process Return Ticket - Issue Refund - Track	- Payment information - User contact information - User Payment information - Items returned for post-returning processing - Return	- Schedule a Pick Up Time requires Store from Choose a Preferred Brick and Mortar Store - Confirm Delivery requires Request Delivery from Make a Purchase - Pay for Delivery requires Payment Information from Enter Payment Information - Enter Delivery Address requires Order Number

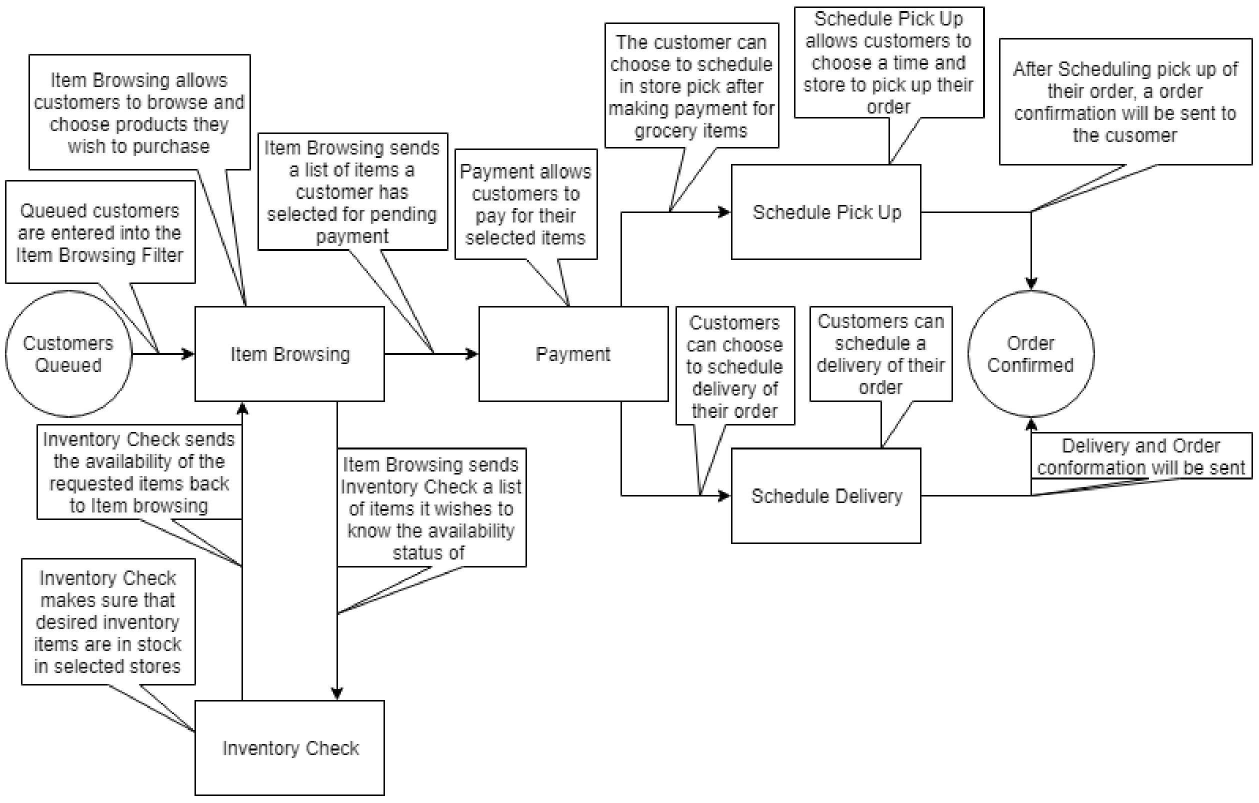
	Previous Purchases - Save Payment Information - View Orders - View Subscriptions	Tickets - Repayment/ Refund history	generated from Make a Purchase
Schedule Pick Up	- Choose Preferred Brick and Mortar Store - Get All Brick and Mortar Stores	- Stores - Pending orders - Delivery payment information - Delivery Address information - Delivery history	- Verify Item Available in Inventory requires Store from Choose a Preferred Brick and Mortar Store - Update Store Inventory requires Items from Add Items to Cart - Update Store Inventory requires Store from Choose a Preferred Brick and Mortar Store - Update Store Inventory requires List of Items Successfully Purchased from Make a Purchase
Schedule Delivery	- Schedule a Pick Up Time - Mark Orders as Picked Up - Pay for Delivery - Enter Delivery Address - Confirm Delivery - Update Pending Orders - Save Delivery Address Information	- Pending orders - Delivery payment information - Delivery Address information - Delivery history - User Delivery information	- Update Store Inventory requires Items from Add Items to Cart - Update Store Inventory requires List of Items Successfully Purchased from Make a Purchase
Inventory Check	- Verify Item Available in Inventory	- Item Inventory - Item View	- Select Item in Store requires Item Inventory from Verify Item Available

	- Update Store Inventory - Update System with new inventory - Check Expiration Date	Counts - Item Purchase Counts - Expiration date of inventory items - User wishlist - Popular products - List/map or similar products - Comparison information of similar products	in Inventory - Add Item to Cart requires Item Can be Added to Cart from Verify Item Available in Inventory - Reorder Subscription required Item Inventory retrieved directly from Inventory where it is allocated - View Popular Products requires Item Purchase Counts retrieved directly from Inventory where is is allocated - View Similar Products requires Item inventory retrieved directly from Inventory where it is allocated - Compare Items required Item Inventory which is retrieved directly from Inventory where it is allocated

Allocation of solutions to deployment components

DB Component	**SB Components Allocated to DB Component**
Item Browsing	Item Browsing
Payment	Payment
Schedule Pick Up	Schedule Pick Up
Schedule Delivery	Schedule Delivery
Inventory Check	Inventory Check

Figure 1: Graphical Depiction of DB1


Item Browsing allows customers to browse and choose products they wish to purchase
Queued customers are entered into the Item Browsing Filter
Item Browsing sends a list of items a customer has selected for pending payment
Payment allows customers to pay for their selected items
The customer can choose to schedule in store pick after making payment for grocery items
Schedule Pick Up allows customers to choose a time and store to pick up their order
After Scheduling pick up of their order, a order confirmation will be sent to the cusomer
Schedule Pick Up
Customers Queued
Item Browsing
Payment
Customers can choose to schedule delivery of their order
Customers can schedule a delivery of their order
Order Confirmed
Inventory Check sends the availability of the requested items back to Item browsing
Item Browsing sends Inventory Check a list of items it wishes to know the availability status of
Schedule Delivery
Delivery and Order conformation will be sent
Inventory Check makes sure that desired inventory items are in stock in selected stores
Inventory Check

Derivation Rationale

Satisfaction of stakeholder qualities (with priorities):

Architectural Style	Stakeholder Needs
Robustness The system is "Robust", minimizing the chances of unexpected crashed or system failures to occur, decreased the chances of system failure, hence preserving the up time of the system, makes it easily adaptable.	*(2) Security* ● The pipe and filter architecture separates client and system data, isolating each to keep confidential information secure. *(1) Reliability* ● This architectural style allows for redundancy and isolates points of failure to ensure maximum reliability. *(9) Scalability* ● The pipe and filter architecture inherently aids scalability as new servers and front ends can be easily added to the broker service. *(8) Evaluability* ● By serving as a central organizer, the broker can accept the evaluations of the front end users and pass it on to the appropriate server.

Simplistic The “Simplistic” nature of the system makes monitoring its components easy, hence, improving the security of the system, makes maintaining or modifying the system easy and efficient, makes testing and evaluations easier to implement.	*(3) Recoverability* ● By using the pipe and filter design within the broker architecture, servers can easily be swapped out in case of failure and system data is isolated to aid in data recovery. *(4) Usability* ● The pipe and filter allows the client code to be completely separate from the server infrastructure, increasing the ease of implementing intuitive user interfaces.
Real Time The “Real Time” nature of the system makes interactions between components fluid and user friendly, creates fast response times and decreased lag between actions.	*(2) Security* ● Dividing our architecture into distinct features isolates sensitive data, minimizing coupling and thus increasing security . *(6) Response Time* ● By increasing component cohesion and decreasing the coupling and thus message communication, this design principle increases performance by streamlining the function pipeline.
Sequential The “Sequential” nature of the system allows for fast detection of errors and ability to quickly fix issues in case of failure, makes adding or modifying component interactions easier as necessary flow paths are easily pinpointed	*(7) Maintainability* ● By ensuring a interface design not based on exact proportions, the system will be easier to maintain as each component is isolated and can be dealt with individually. *(10) Adaptability* Since this design principle defines the interfaces and forces components to act independently, the system can be more adaptable as features become more plug and play.

Conflicts and Refactoring

In its current form, the DB design does not pose any conflicts with itself nor the BB, and does not refactor from the original BB design.

Design Inspired By (Style/Guideline References):

- Pipe-and-filter architecture: Master records are "transformed" at each step in the process.
- Domain realities: Deployment components closely mirror business blueprint components.

References

- Pipe-and-filter: http://www.dossier-andreas.net/software_architecture/pipe_and_filter.html

Deployment Blueprint #2 (DB2) – Client/Server Solution

Satisfaction of domain functions, data and I/O by solutions

SB Solution Component	**BB Functions Satisfied**	**BB Data Satisfied**	**I/O Satisfied**
Server Authentication Module	- Authenticate User - Authenticate Employee - Authenticate Administrator - Enable Two-Factor Authentication	- Employee ID - User ID - Administrator ID - Passwords - Accounts - Login Sessions - Two-Factor Authentication Enable - User Account - Employee Account - Administrator Account	- Track Previous Purchases requires User Account retrieved directly from Administration where it is allocated - Send Notification requires User Account retrieved directly from Administration where it is allocated - Add Item to Wishlist requires User Account retrieved directly from Administration where it is allocated
Server Delivery Module	- Schedule a Pick Up Time - Mark Orders as Picked Up - Pay for Delivery - Enter Delivery Address - Confirm Delivery - Update Pending Orders - Save Delivery Address Information	- Order - Pending orders - Delivery payment information - Delivery Address information - Delivery history	- Confirm Delivery requires Request Delivery from Make a Purchase - Pay for Delivery requires Payment Information from Enter Payment Information - Enter Delivery Address requires Order Number generated from Make a Purchase
Server Pick-Up Module	- Choose Preferred Brick and Mortar	- Stores - User Subscriptio	- Schedule a Pick Up Time requires Store from Choose a

	Store - Get All Brick and Mortar Stores	ns - Order History - List of Orders the user has placed - Items from Past Order - User Payment information - User Delivery information - User Subscriptions	Preferred Brick and Mortar Store - Verify Item Available in Inventory requires Item from Add Item to Cart - Verify Item Available in Inventory requires Store from Choose a Preferred Brick and Mortar Store - Update Store Inventory requires Items from Add Items to Cart - Update Store Inventory requires Store from Choose a Preferred Brick and Mortar Store - Update Store Inventory requires List of Items Successfully Purchased from Make a Purchase
Server Payment Module	- Enter Payment Information - Send Notifications - Schedule Refund - Process Return Ticket - Issue Refund - Track Previous Purchases - Save Payment Information - Save Delivery Address Information - Edit or Delete Subscriptions	- Payment information - Confirmation/Order Numbers - Items successfully purchased - Subscription orders - User contact information - Items returned for post-returning processing - Return Tickets - Repayment/Refund history	- Schedule Refund requires Item from Past Order from View Orders - Issue Refund requires User Payment Info which is retrieved directly from User Data where it is allocated - Process Return Ticket requires Payment Information which is generated from Enter Payment Information - Reorder Subscriptions requires User Subsctriptions which is allocated in User Data
Server Main Browsing	- Select Item in Store	- Item Selected	- Check Expiration Date requires Selected Item

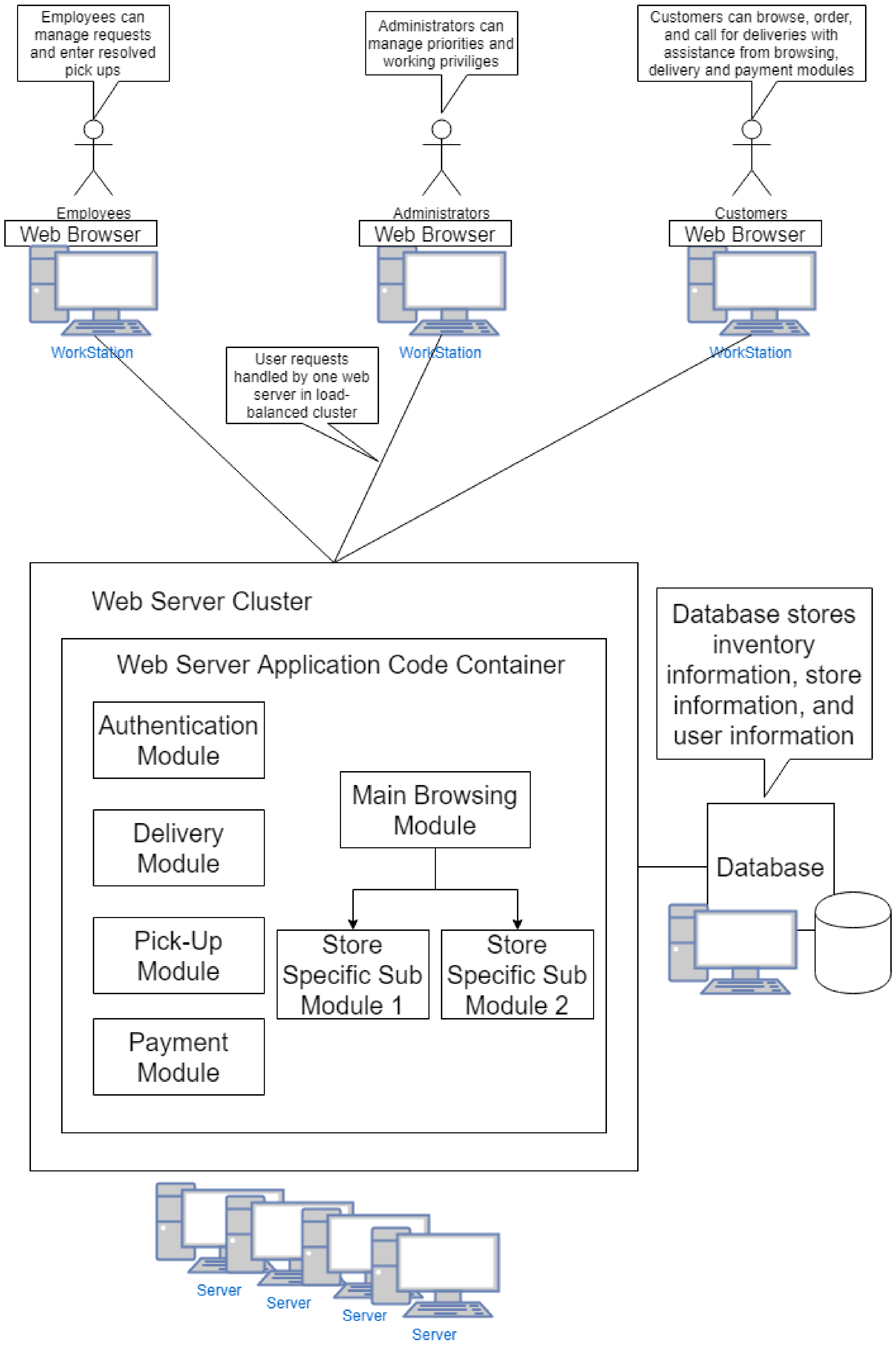
Module	- Add Item to Cart - Make a Purchase - Reorder Subscription - Create Subscription - Verify Item Available in Inventory - Update Store Inventory - Update System with new inventory - Check Expiration Date - Add Item to Wishlist - View popular Products - View Similar Products - Compare Items - View Orders - View Subscriptions	- Shopping Cart - Item Inventory - Item View Counts - Item Purchase Counts - Expiration date of inventory items - User wishlist - Popular products - List/map or similar products - Compariso n information of similar products	from Select Item in Store - Add Item to Wishlist requires Selected Item from Select Item in Store - Compare Items required Items which is retrieved directly from Shopping where it is allocated - iew Popular Products requires Item Inventory retrieved directly from Inventory where is is allocated - View Popular Products requires Item View Counts retrieved directly from Inventory where is is allocated - View Popular Products requires Item Purchase Counts retrieved directly from Inventory where is is allocated - View Similar Products requires Item inventory retrieved directly from Inventory where it is allocated - Compare Items required Item Inventory which is retrieved directly from Inventory where it is allocated

Allocation of solutions to deployment components

DB Component	**SB Components Allocated to DB Component**
Web Server Cluster	Server Authentication Module
Web Server Cluster	Server Delivery Module
Web Server Cluster	Server Pick-Up module
Web Server Cluster	Server Payment Module

Web Server Cluster	Server Main Browsing Module
Database	N/A
Web Browser	N/A

Figure 2: Graphical Depiction of DB2


Employees can manage requests and enter resolved pick ups
Administrators can manage priorities and working priviliges
Customers can browse, order, and call for deliveries with assistance from browsing, delivery and payment modules
Employees
Web Browser
WorkStation
Administrators
Web Browser
WorkStation
Customers
Web Browser
WorkStation
User requests handled by one web server in load-balanced cluster
Web Server Cluster
Web Server Application Code Container
Authentication Module
Delivery Module
Pick-Up Module
Payment Module
Main Browsing Module
Store Specific Sub Module 1
Store Specific Sub Module 2
Database stores inventory information, store information, and user information
Database
Server
Server
Server
Server

Derivation Rationale

Satisfaction of stakeholder qualities (with priorities):

Architectural Style	Stakeholder Needs
Centralized Nature The system's "centralized nature" makes it reliable and resilient to unexpected errors. Web server "farm" produced high throughput.	*(9) Scalability* ● The client server style allows for easy addition of more clients to the system through the use of server side optimizations, including load balancing. (4) *Usability* ● Using this design style allows the client to introduce intuitive design into the system by modifying the client only, pulling data from the server separately. *(8) Evaluability* ● Because there is a central server to send evaluations to, this architecture design aids evaluability by letting the user pinpoint gui changes which can be used to correspond to the server component. *(10) Adaptability* ● Separating the client from the server allows the store to change or modify the client without changing the underlying functionality available.
Automated Checkers Server subsystems are supported by automated checkers that are subclasses implementing abstract methods of the parent main system class, the subclasses are easily accessible for security risks.	*(3) Recoverability* ● The central database ensures all the data can be backed up and restored rapidly, thus aiding the recovery time in case of failure.
Record Locking in Database System records are locked by database, but transaction time is short, so current data is rarely unavailable for viewing, making transaction time is short, so current data is rarely unavailable for viewing.	*(5) Availability* ● Having a components separated allows each component to exist separately, increasing the amount of services available even if one fails due to increased independence. *(6) Response Time* ● Promoting object oriented frameworks increases performance by keeping all the data and functions which are atomic together within a component. Thus it reduces

	unnecessary connections and minimizes latency. *(7) Maintainability* ● Object oriented components are easy to swap and maintain appropriately as they are contained services.
Use of open shelf /open source technologies Initial hardware will be provided mostly through off the shelf or open source, making maintenance very easy.	*(2) Security* ● Separating components into components based on the type of service they are increases the security of the system by protecting and isolating sensitive data. *(1) Reliability* By abstracting each type of service into a component, the points of failure are minimized into types, increasing reliability by making each service easier to debug and observe.

Conflicts and Refactoring

In its current form, the DB design does not pose any conflicts with itself nor the BB, and does not refactor from the original BB design.

Design Inspired By (Style/Guideline References):

- Client-server architecture: Need for a central student record and course database.
- Object-oriented architecture: Inheritance used for specialized rule-checkers.
- Web server redundancy: Web server requests distributed across multiple servers in cluster.

References

- object-oriented architecture: http://people.cs.uchicago.edu/~mark/51050/lectures/lecture.1/lecture.1.pdf
- client-server architecture: http://en.kioskea.net/contents/cs/csintro.php3
- web server redundancy: http://www.westwind.com/presentations/loadbalancing/networkloadbalancingwindows2003.asp

Part 2: Evaluating Solution Blueprint Compliance

Compliance Metrics for DB1

DB Component	SB Component	CompFuncCoeff (c,t)	CompDataCoeff (c,t)	CompFuncBoundaryError(t)
Item Browsing	Item Browsing	0.70	1.00	0.30
Payment	Payment	1.00	0.83	0.00
Schedule Pick Up	Schedule Pick Up	1.00	0.80	0.00
Schedule Delivery	Schedule Delivery	0.71	0.80	0.29
Inventory Check	Inventory Check	1.00	0.88	0.00

Compliance Metrics for DB2

DB Component	SB Component	CompFuncCoeff (c,t)	CompDataCoeff (c,t)	CompFuncBoundaryError(t)
Web Server Cluster	Server Authentication Module	0.75	1.00	4.39
Web Server Cluster	Server Delivery Module	0.83	0.80	4.39
Web Server Cluster	Server Pick-Up module	1.0	0.88	4.39
Web Server Cluster	Server Payment Module	0.88	0.88	4.39
Web Server Cluster	Server Main Browsing Module	0.93	1.00	4.39

Part 3: Planning Evaluation of Deployment Blueprints Using an ATAM Quality Attribute Utility Tree

Aggregation Discussion

Aggregation in this model will be normalized into a “metric scale” such as “good” = 3, “acceptable” = 2, and “poor” = 1. The rating of each objective will be scaled by the priority of its classification in a model (10 - Priority) + Metric Rating. For example, if security earns a score of 2 on its metric scale, it will contribute (10 - 2) * 2 = 16 points to the overall aggregate rating. In this model, total aggregate rating will be out of a maximum of 165 points. For reliability, a crash rate of less than once a year is considered “good”, once a year is “acceptable”, and more than once a year is “poor”. For security a breach rate of less than once a year is considered “good”, once a year is “acceptable”, and more than once a year is “poor”. For recoverability, recovery time of less than 1 minute is considered “good”, less than 5 minutes is “acceptable”, and anything longer is “poor”. For usability, a crash rate of less than once a year is considered “good”, once a year is “acceptable”, and more than once a year is “poor”. For availability, a crash rate of less than once a year is considered “good”, once a year is “acceptable”, and more than once a year is “poor”. For response time, a transaction response time under 2 seconds is "good," while under 3 seconds is acceptable and anything greater than 3 seconds is unacceptable. For maintainability, maintenance time of under 30 minutes if “good”, under 1 hour is “acceptable”, and anything over is “poor”. For evaluability, evaluation time of under 30 minutes if “good”, under 1 hour is “acceptable”, and anything over is “poor”. For scalability, scaling time or under 30 minutes if “good”, under 1 hour is “acceptable”, and anything over is “poor”. For adaptability, anything under 30 minutes if “good”, under 1 hour is “acceptable”, and anything over is “poor”.

ATAM Tree

- UTILITY
    - **Reliability**
        - <u>Invest in backup servers</u>
            - *Objective:* Improve robustness and allow customers to browse product catalog, add items to cart, and successfully complete transactions (H,M)
                - *Metric:* Number of crashed or unsuccessful actions that were mitigated through the investment in backup servers (M)
    - **Security**
        - <u>Enable two-factor authentication</u>
            - *Objective:* Improve protection of sensitive store and customer data from hacking while also maintaining a secure payment system for the brick and mortar stores (H,M)
                - *Metric:* Number of attacks that were diverted or nullified by the enablement of two factor authentication (M)
    - **Recoverability**
        - <u>Ensure record integrity</u>
            - *Objective:* Maximize recoverability from database corruption or deletion with minimal loss of productivity (H,H)
                - *Metric:* Number of records that were able to be recovered due to the measures assumed (L)
    - **Usability**
        - <u>Enable user feedback</u>
            - *Objective:*Maximize familiarity by hosting platforms familiar to users, while maintaining a desired workflow that should be intuitive (M,M)
                - *Metric:* Amount of feedback that were received as a result of the enablement of user feedback (L)
    - **Availability**
        - <u>Invest in cloud servers</u>
            - *Objective:* Improve the time the system should always be up and available, except for rare, appropriate circumstances (M,H)
                - *Metric:* Amount of additional uptime that were the result of the investment of cloud servers (H)
    - **Response Time**
        - <u>Minimize transaction response time</u>
            - *Objective:* Improve the times the system should offer low latency for all user requests over common network speeds (M,M)
                - *Metric:* Average improved response time as a result of actions taken to improve response time (H)
    - **Maintainability**

        - Maximize monitoring of system functions
            - *Objective:* Enhance the system's ability to be maintained easily and quickly, minimizing downtime (L,M)
                - *Metric:* Amount of system functions successfully monitored by the prescribed actions (H)
- **Evaluability**
    - Maintain system traces
        - *Objective:* Maximize the system's ability be able to be evaluated and critiqued, by both customers and employees (L,L)
            - *Metric:* Amount of system traces that were monitored as a result of prescribed actions (M)
- **Scalability**
    - Maximize scalability
        - *Objective:* Improve the system's availability to be easily extended to support future brick and mortar stores, a larger product catalog, and a larger user base in the case of widespread adoption. (L,M)
            - *Metric:* Difficulty of scaling the system to new functionalities (H)
- **Adaptability**
    - Enable Pluggable advising service
        - *Objective:* Maximize the system's ability to adapt to specific themes and cultures of specific companies or holidays and seasons (L,H)
            - *Metric:* Ease of which the system is able to adapt specified themes and cultures (L)

**H = High, M = Medium, L = Low
**(H, L) indicates the objective is of high importance but is hard to achieve
**(H) indicates the data/statistic accurately predicts/assesses the objective satisfaction