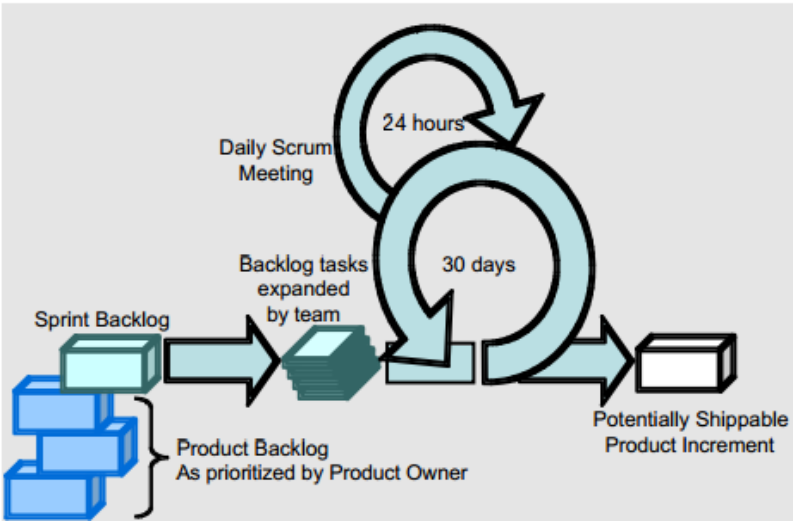
Web Engineering Exam Notes

1. Software process and web application architecture

Present your web application engineering process and discuss your choices.

SW Process - Scrum

Product Backlog, Sprint, Sprint Planning Meeting, Sprint Backlog, Roles - Product Owner, Scrum Master, Scrum Team, Daily Scrum Meeting, and Sprint Review Meeting



Web Engineering Process

WebML [Web Modelling Language] - (Similar to Waterfall)

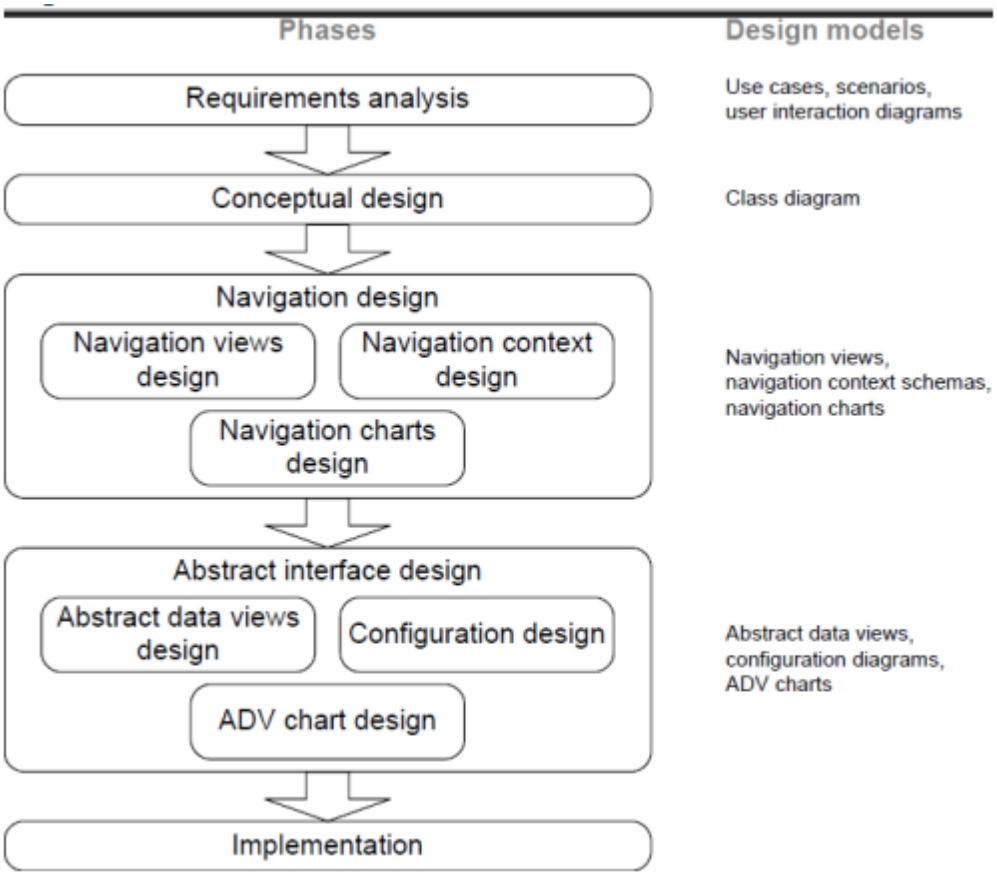
Usual development process
Requirements -> Design (Data & Hypertext) -> Implementation -> Test -> Deployment

WSDM [Web Site Design Method]

Focus on the design - Know your customer and design according to the customer
The Mission -> Customer modelling -> Conceptual Design -> Implementation Design -> Implementation

OOHDM [Object-Oriented Hypermedia Design Method]

Focus on the product (opposed to the customer) - How is the information easily structured
Requirements -> Conceptual design -> Navigation Design -> Interface Design -> Implementation



OOHDM and Scrum

Sprints are done as OOHDM - Because it focuses on how the information is displayed
Each sprint -- Requirements -> Design -> Implementation

Discuss the advantages and disadvantages.

- Pros
  - Adaptable to changes
  - Choose work tasks (ownership)
  - Result after each sprint
  - Focus on displaying the information (fx. how to navigate)
- Cons
  - Difficult to develop with a traditional minded customer
  - Not so much focus on usability

Present it on your own application case or your own project case. Present what are possibilities for tiers of the web application and how they relate to technology.

.NET

The Microsoft MVC architecture supported in Microsoft .NET framework
Advantage

- We are all familiar with .NET, however at varying degrees - Everyone has developed a

web application at 3rd semester
- Supported by a good IDE - Visual Studio
- OOP

Disadvantage
- Microsoft Server is the Nativedb

PHP

Web development with HTML, JavaScript, and PHP
Advantage
- It is fast getting started developing a web application in PHP.

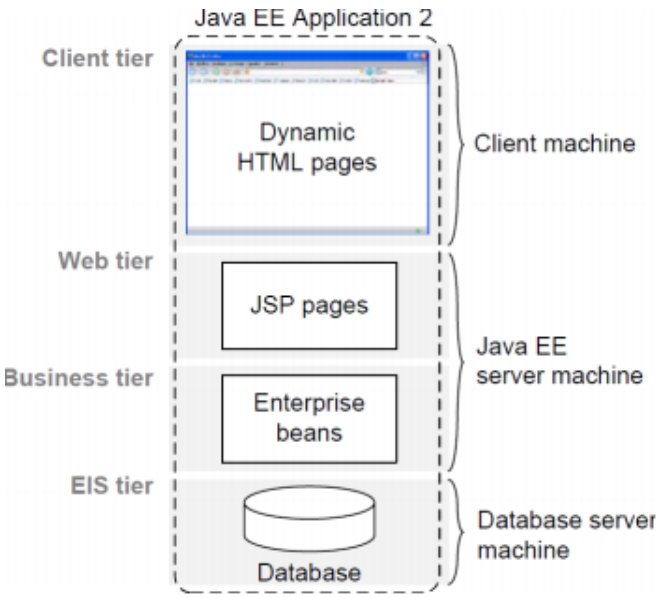
Disadvantage
- Not everyone in the group is familiar with PHP
- There are many security concerns that have to be addressed when developing in PHP, partially due to PHP being a weakly typed language.

Java EE

Java Platform (Java EE) on the application server.
Advantage
- Everyone is familiar with Java
- OOP

Disadvantages
- There is a large overhead to setting up a layered structure using Java.
- No one has previously developing a Java web application.

Architecture - MVC



The Java EE application architecture
- Partition the application into -- Views, Controllers and Models

Discuss your decision on your own project or application. Discuss advantages and disadvantages.

Advantage

- More unit testable
- No business logic in the UI
- Easier to maintain, extend, and reuse

Disadvantage

- More initial work (more code)

2. Data I: XML

Present your approach to load an XML file.

XSLT [eXtensible Stylesheet Language Transformations]

A Stylesheet and transformation language.
Transforms XML data into HTML or other XML by using XSLT

Example of a xsl template, which

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html" indent="yes" encoding="UTF-8"/>
  <xsl:template match="/">
    <html xmlns="http://www.w3.org/1999/xhtml">
      <head>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
        <title>Ebay Seller Info</title>
      </head>
      <body>
        <table>
          <tr>
            <th>Seller Name</th>
            <th>Seller Rating</th>
          </tr>

          <xsl:for-each select="root/listing/seller_info">
            <tr>
              <td><xsl:value-of select="seller_name"/></td>
              <td><xsl:value-of select="seller_rating"/></td>
            </tr>
          </xsl:for-each>
        </table>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

Discuss approaches on how to present subset of the file in a client structured presentation (could be a table or structured lists).

Tables for detailed specifications and comparisons
Lists for product overviews

Reflect on what you have learn, on difficulties, on shortcomings and advantages of XQuery and XSLT and what from those technologies you have used for your implementation and why.

Difficulties

- Difficult to debug

Advantages

- XQuery is similar to PHP
- XSLT is similar to XML
- Well documented languages (w3.org approved)

XQuery

XQuery is for XML what SQL is for Databases.

Neither javascript itself or any webbrowser (IE, FF, Opera, ...) are XQuery capable or have support build in. You need some other software which has XQuery implemented.
This means that it is easiest to make the query server side.

Example of Query XML documents directly in a HTML page -- eBay listings:

```
<html><head/><body>
 {
  for $ebay in doc("ebay.xml")
  let $listings := distinct-values($item//LISTING)
  return
    <div>
      <h1>{ string($listings/NAME) }</h1>
      <ul>
      {
        for $listing in $listings
        return <li>{ $listing }</li>
      }
      </ul>
    </div>
 }
 </body></html>
```

XQuery	XSLT	DOM
Dynamic	Templates	Dynamic
HTML	XML/HTML/RDF	HTML/Java Objects

Discuss also in comparison to RDF (Resource Description Framework)

XSLT can be used to create a RDF from an XML file.
RDF adds metadata

3. Data II: Semantic Web Technologies

Present RDF and RDFS model on your implementation application.

RDF = Resource Description Framework

RDF can be used to describe relations or metadata on the web, in RDF you can create an object as a resource and then reference it.

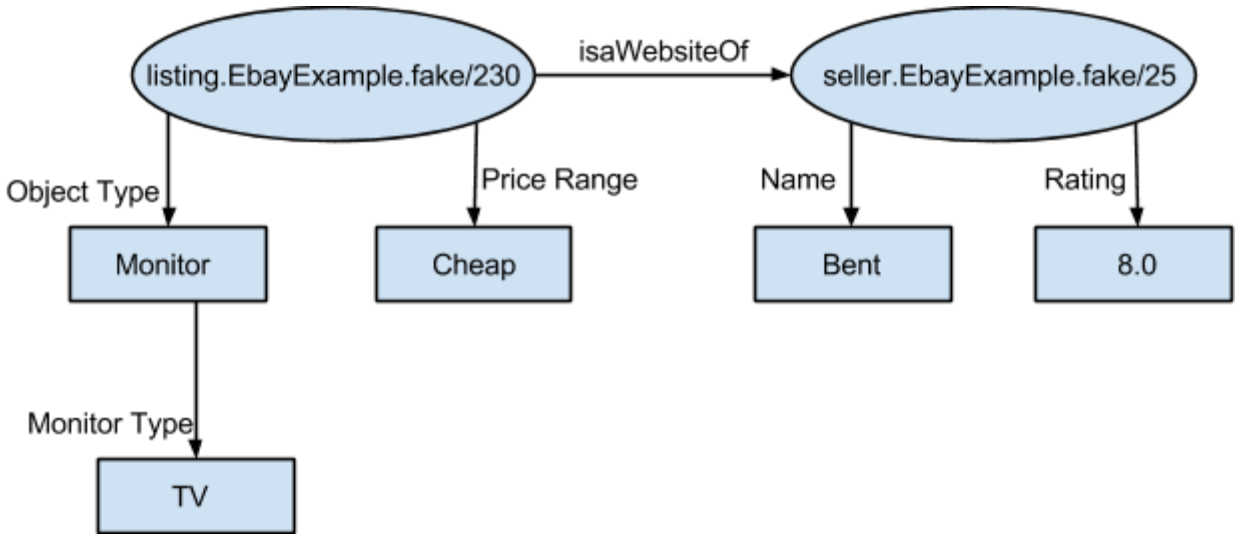
SPARQL is then used to query in RDF schemas.

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:ebayurl="http://www.ebayexample.fake#">
      <rdf:Description
      rdf:about="http://www.ebayexample.fake/index">
            <ebayurl:isaWebsiteOf rdf:resource="ebay"/>
      </rdf:Description>
      <rdf:Description rdf:about="ebay">
      <rdf:type
      rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
      </rdf:Description>
</rdf:RDF>
```

RDF validator: http://www.w3.org/RDF/Validator/

RDF can be used to model the structure of websites.

A small example of our RDF model for our implementation:



Discuss it also in the context of the transformation you needed to make from the XML data.
A little data transformation is needed because of the differences in RDF and XML. RDF is structured in tuples where the start and end point can be resources and XML has a hierarchical tree structure with a root element which can lead down to all other sub elements in the tree.
Reflect on RDF, RDFS and the technology you have used also in connection to XML.

Present your experience with SPARQL and the main concept of the SPARQL in the context of the dbpedia sparql web interface and find related data to your XML exercise.

SPARQL [SPARQL Protocol and RDF Query Language]
SPARQL is used to query on RDF written in XML in the same way as SQL is used to query a database.
SPARQL query example for the EbayExample:

prefix ebay: <http://ebayExample.fake/data/>
select ?seller_name
where { ?listing ebay:name ?seller_name . }
Resource Property Propertyvalue

Present part of the data and discuss on them advantages, disadvantages of RDF/RDFS and SPARQL also in connection to XML.
Pros
- Enables users to find all relevant data from repositories
- Data can be found across multiple website or repositories
- Merging of results it possible
- Easier for search engines to index websites with RDF

Cons
- Data must be accessible
- A lot of the web must be converted to RDF
- For storage only, the metadata in RDF is irrelevant

If you do not find the data relevant to your XML data set, make some examples of connections between dbpedia data and your XML data to show the connections and how they can be made.

4. Architecture and Patterns

Take one pattern from each layer (model, view and controller) and present its implementation in your application.

- Domain Model
    - Domains handling subjects of that domain

```
public class Listing {

	private Seller seller;
	private Item item;
	private List<Bid> bids;

	public Listing(Seller seller, Item item){
		this.seller = seller;
		this.item = item;
	}

	public Seller getSeller() {
		return seller;
	}

	public void setSeller(Seller seller) {
		this.seller = seller;
	}
}
```

- Template View
    - Markers in a HTML page to insert information (Like PHP)

```
  <ui:param name="pageTitle" value="New Bid" />

    <ui:define name="main">
      <h:form id="form">
           <p:panel header="" style="margin-bottom:10px;">
               <p:messages id="messages" showDetail="true"
               autoUpdate="true" closable="true" />


                <h:outputLabel for="bid" value="Bid:*" />
                    <p:inputText id="bid"
                    value="#{ListingController.listing.bid}"
                    required="true" />
```

```
                    <p:commandButton value="Submit" update="msgs"

                actionListener="#{ListingController.bid}"
                id="btnSubmit" />
            </p:panel>
        </h:form>
    </ui:define>
```

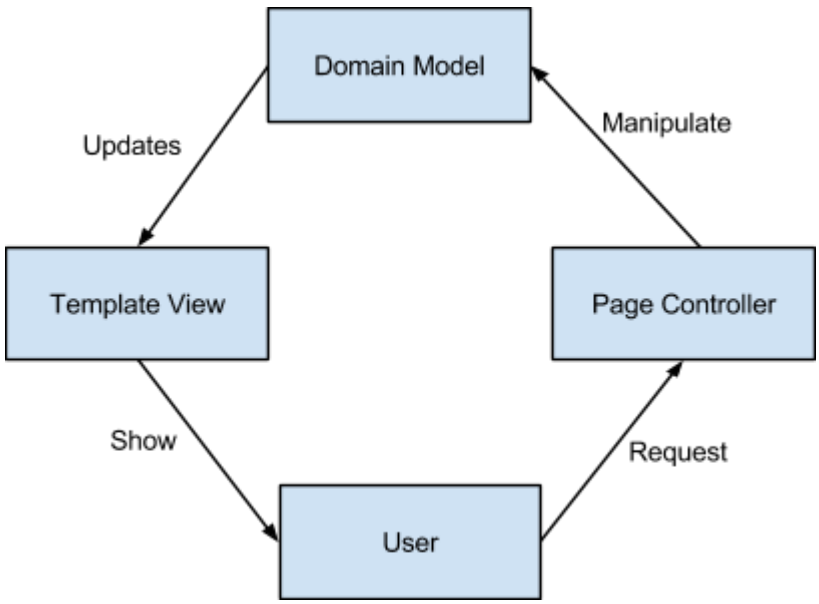
- Page Controller
    - One controller for each page

```
public class ListingController {
      private Listing listing;
      private Bid bid;

      public void Bid(){
            listing.addBid(bid);
      }

      public Bid getHighestBid(){
            return listing.getHighestBid();
      }
}
```

Discuss how you have implemented that.



Reflect on its advantages and disadvantages.

- Pros/Cons
    - A lot of control
    - Might give some overhead because of the page controllers

Expand on the discussion of advantages and disadvantages about some selected other patterns which you have not implemented in your assignment.

- What could have been done
    - Transaction Script -- No need for many different transactions, only handle bids
    - Template View -- Use XQuery or XSLT to parse the XML in the pages
    - Front Controller -- Redirecting the user to e.g. eBay if the listing has ended

- MVC - patterns briefly
- Model - Information about the domain, with non-visual data and behavior.
    - Transaction Script -- Small script handling DB
    - Table Module -- One layer handling all transactions, entire entries at the time
    - Selection: Domain Model -- Domains handling subjects of that domain
- View - UI only about displaying the data.
    - Transform View -- Transforms a domain into a view
    - Selection: Template View -- Markers in a HTML page to insert information (Like PHP)
- Controller - Manipulates data from the model and handles data transfer to the view.
    - Application Controller -- Wizard like, step by step, gather information about previous actions
    - Front Controller -- One controller handling all requests, determine action from URI
    - Selection: Page Controller -- One controller for each domain

5. Design: Navigation and Presentation

Present the design and implementation of a navigation and presentation for some of the data you have processed in the XML assignment in your application.

```
<h:form>

<p:growl id="msg" showDetail="true" />

<p:tagCloud model="#{tagCloudBean.model}">
    <p:ajax event="select" update="msg" listener="#{tagCloudBean.onSelect}" />
</p:tagCloud>

</h:form>
```

```
@Named("TagCloud")
public class TagCloudBean {

    private TagCloudModel model;

    public TagCloudBean() {
        List<EbayListing> listings = XmlParser.getListings();

        model = new DefaultTagCloudModel();

        for (EbayListing l : listings) {
            HashMap<String, Integer> tags = getTags(l.desc);
            for (String tag : tags.keySet()) {
                model.addTag(new DefaultTagCloudItem(tag, tags.get(tag)));
            }
        }
    }

    private HashMap<String, Integer> getTags(String desc) {
        .
        .
        .
    }

    public TagCloudModel getModel() {
        return model;
    }

    public void onSelect(SelectEvent event) {
        TagCloudItem item = (TagCloudItem) event.getObject();
        FacesMessage msg = new FacesMessage(FacesMessage.SEVERITY_INFO, "Item Selected", item.getLabel());
        FacesContext.getCurrentInstance().addMessage(null, msg);
    }
}
```

Discuss why you have chosen particular design technique and particular design and reflect on its advantages and disadvantages.

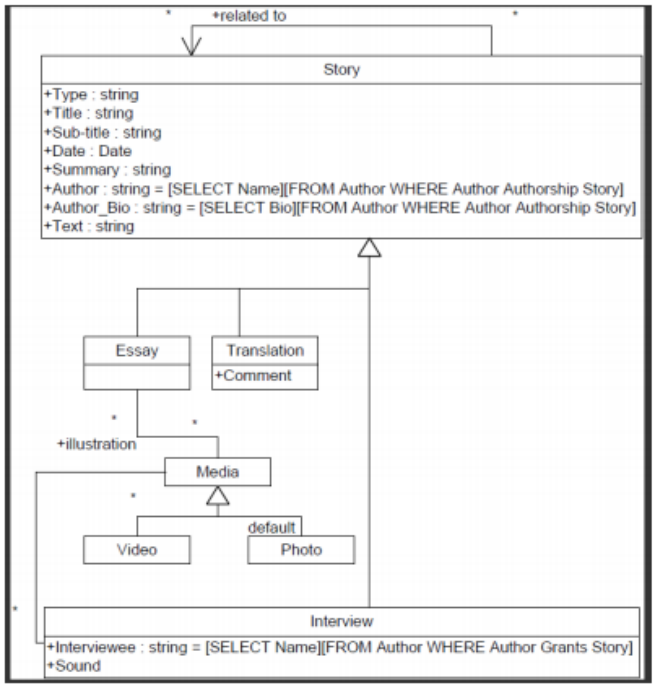
A tag cloud limits the users possibilities to find items that are not popular.

Discuss and reflect on relation between design and implementation.

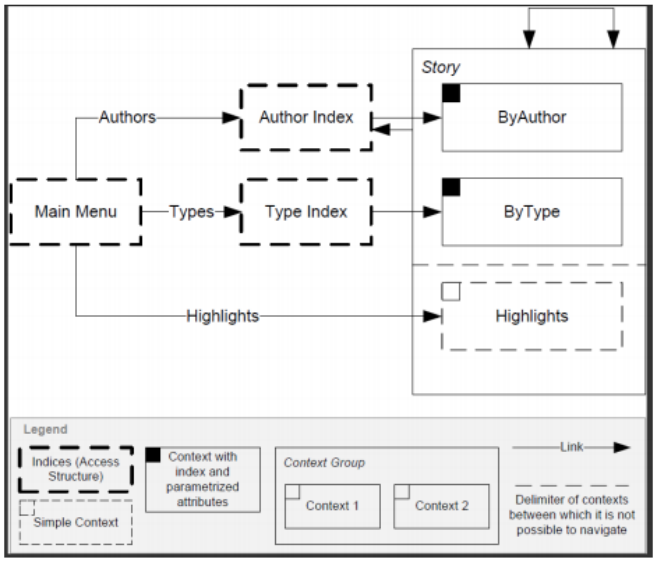
- *Navigation - Description*
    - *Site Structure -- How navigation nodes are defined and linked together*
        - *OOHDM [Object Oriented Hypermedia Design Method]*
            - *Navigation Views -- Relation, custom SELECT statements*

(Author Authorship Story)

- *eBay - Describes relation between an item and its sellers/buyers/prices*

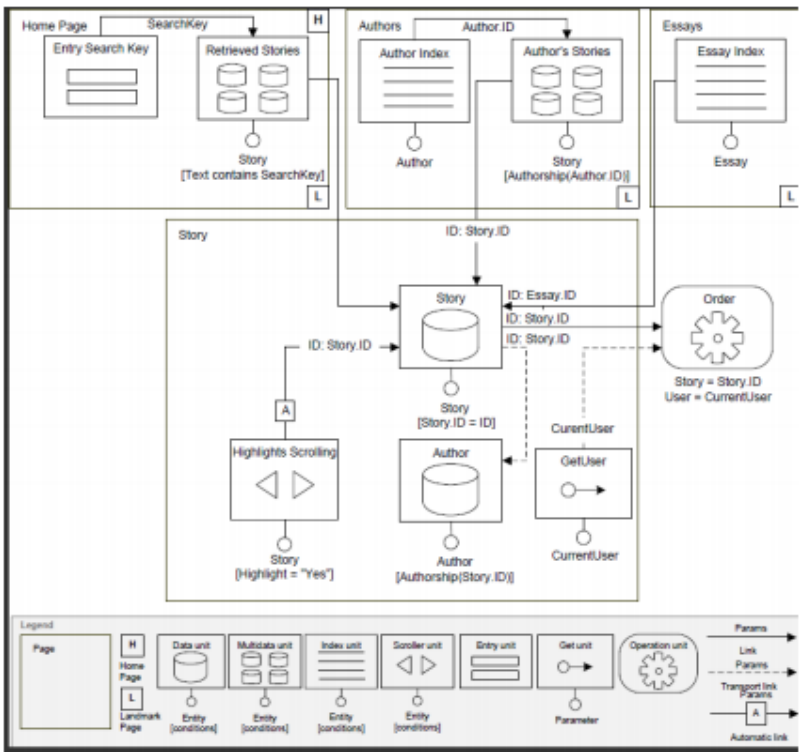


- *Navigation Schema -- Navigation Structure and mechanism*
  - *Describes how the website can be navigated*
  - *eBay - Object type, Price, Popular*

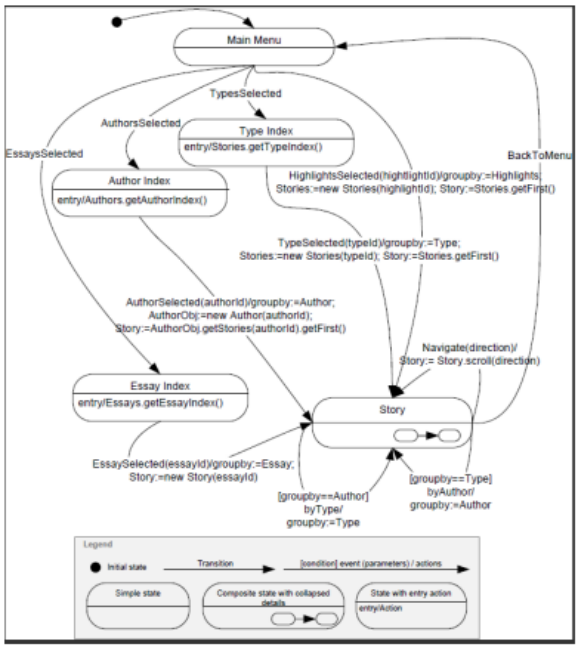


- *InContext Class -- Enriches the objects, more details than the views*

- *WebML -- Describes exactly how the pages should interact*



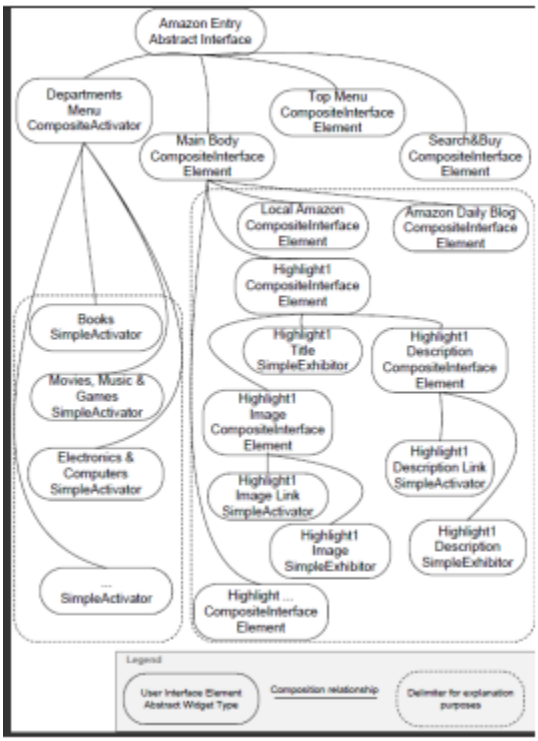
- *Navigation behaviour -- How the navigation works and what it triggers*
  - *UML Guide - Interactions and triggers*



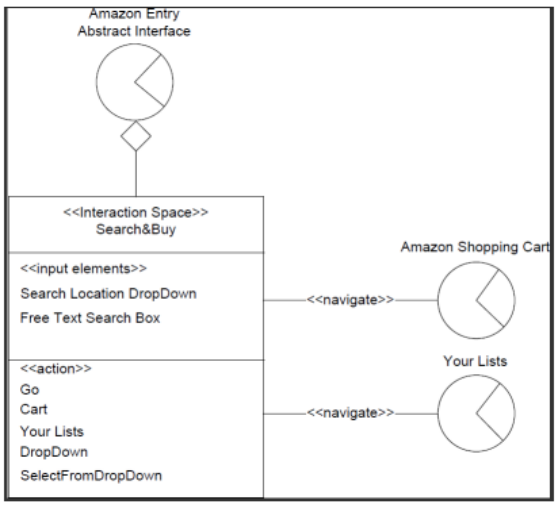
  - *Implemented navigation - why?*
    - *OOHDM and UML*
      - *Because they give a good overview in diagrams that all developers know -- possibilities to outsource*
      - *There is no questions of what the objects should include*

- *Presentation - Description*

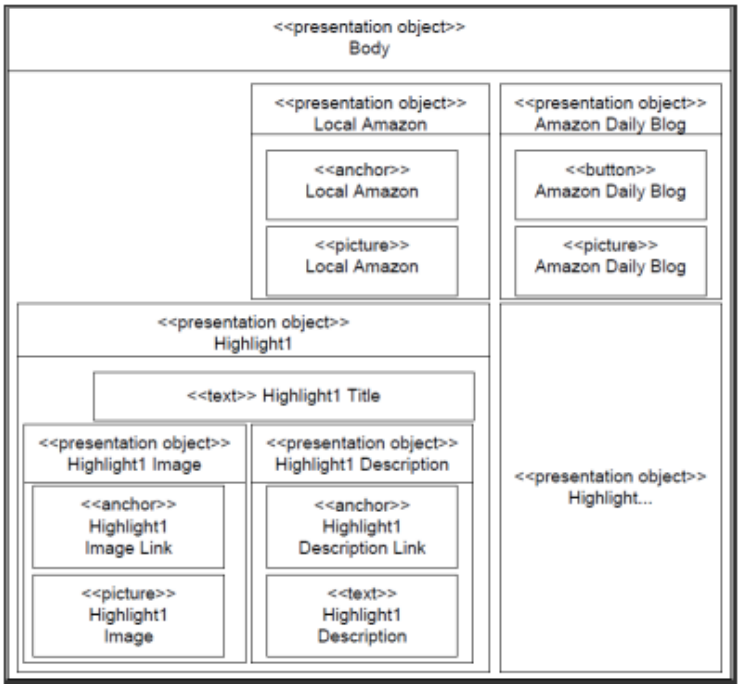
- *Abstract Widgets Design*
  - *UI as composition of widgets that represent classes of elements*



  - *Interaction space of a component*



- *Concrete Presentation Design*



    - *Choice of presentation - Combination of abstract and concrete presentation*
        - *The abstract widgets, to describe functionalities and interactions*
        - *Concrete to describe placements of the widgets*
- *Relation between design and implementation*
    - *We have described the classes, implemented directly*

6. Requirements

Think in wider context of your web application built gradually in previous assignments. Discuss what goals it achieves or can participate in, discuss which value exchanges it can provide, which workflows it can fit in, and in which information exchange it can participate in.

Achieved Goals:

- Import eBay data through XML to the web site
- RDF to search relations between objects in the website
- Navigate between eBay listings
    - According to attributes

Value Exchange

- Referencing customers to eBay
- Bid on items listed

Workflow to fit in

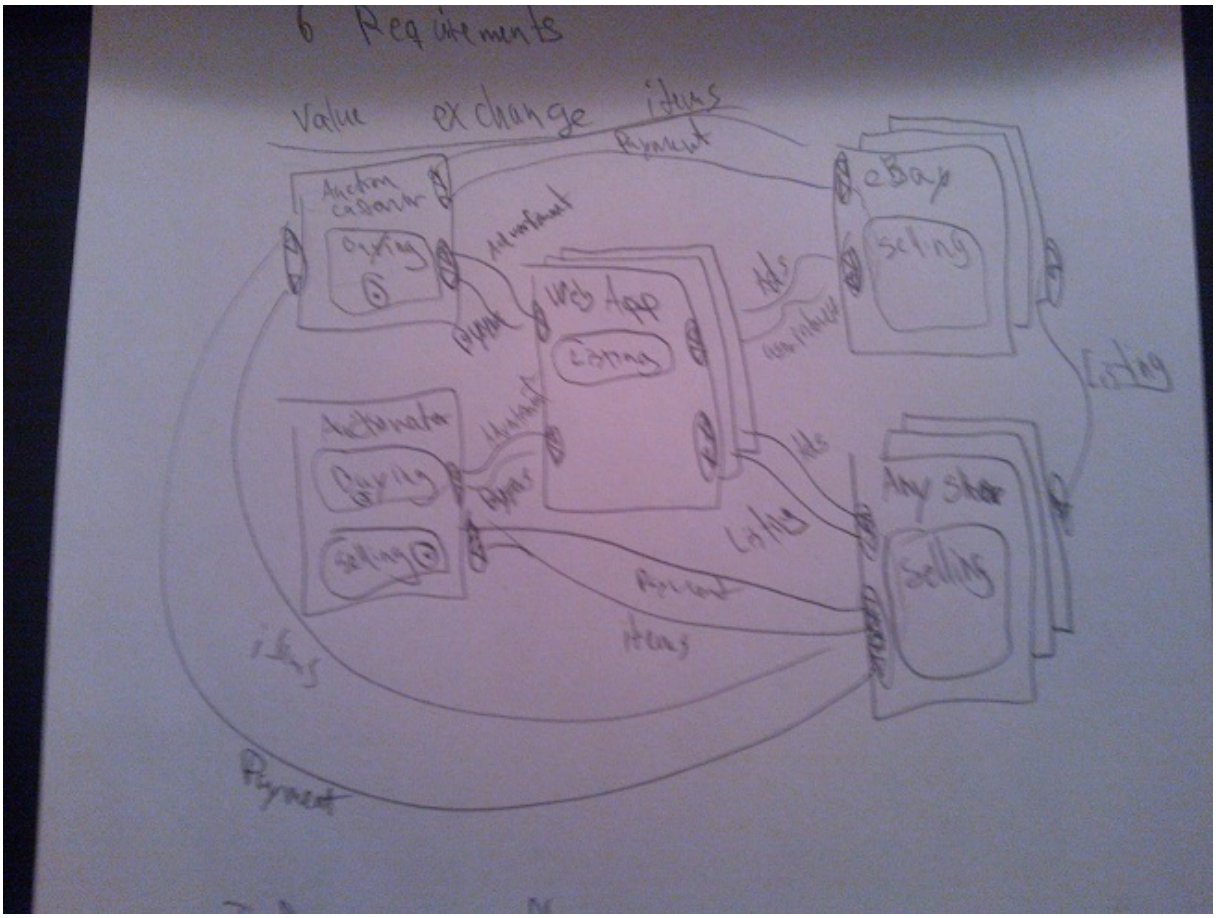
- When looking for hardware auctions on eBay
- Paying for bids on eBay listings

Information Exchange

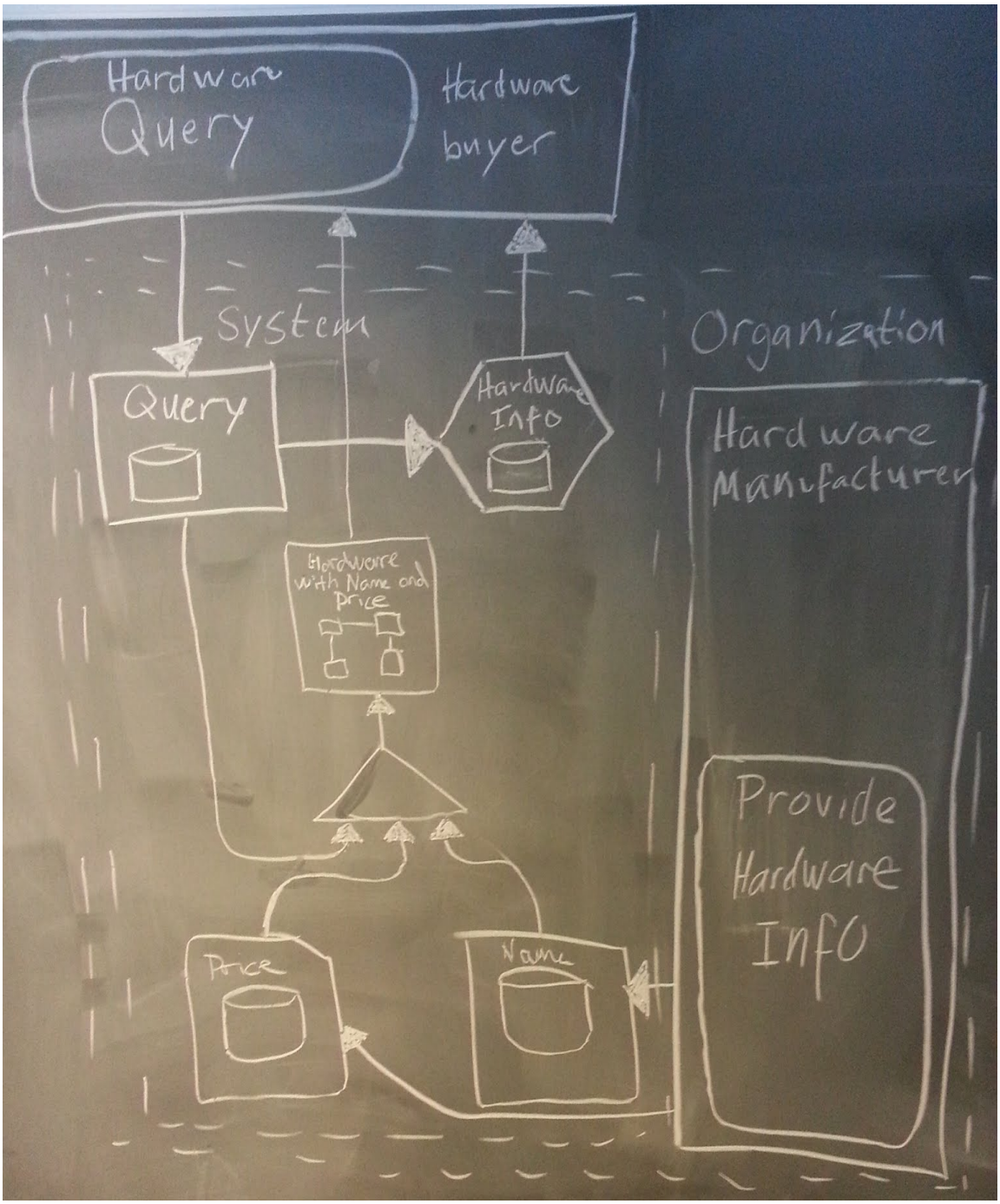
- Find eBay listings according to current highest bid
- Find eBay listings according to requirements
- Find items descriptions
- Credit card information when bidding
- Personal information when bidding

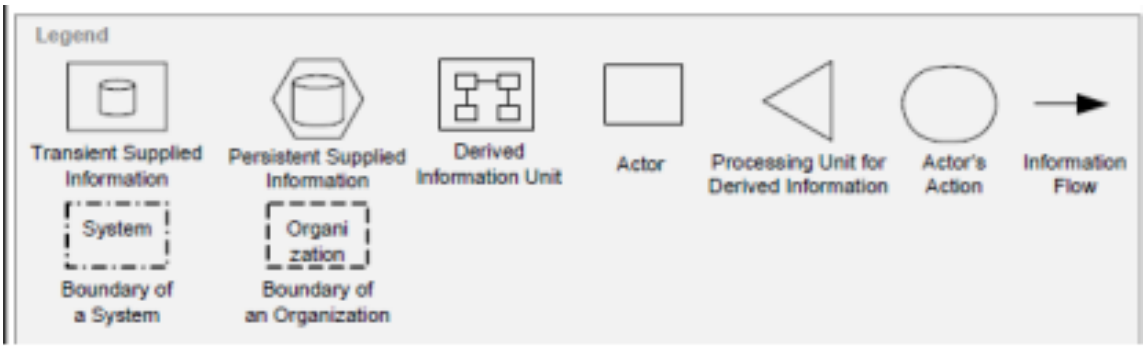
Show it on the techniques for requirements engineering in web application context.

Value Exchange Requirements Engineering



Information Flows Requirements Engineering


Hardware Query
Hardware buyer
System
Organization
Query
Hardware Info
Hardware with Name and price
Hardware Manufacturer
Provide Hardware Info
Price
Name


Legend
Transient Supplied Information
Persistent Supplied Information
Derived Information Unit
Actor
Processing Unit for Derived Information
Actor's Action
Information Flow
System
Boundary of a System
Organi zation
Boundary of an Organization

Discuss advantages and disadvantages of the technique in your context and also in general.

Pros

- Gives an easy overview of the plan with the website

Cons

- Takes time to manufacture and understand how to use it

7. Quality I: performance and dataflow testing

Suggest and discuss how and why you would test the reliability and performance of your application you have implemented in your assignments.

Reliability and Performance - Description

- Throughput -- Number of requests per unit of time
- Response time -- Time to server a request
- Is there a need to upgrade hardware or software?
    - Don't want customers to wait when they bid
    - Don't want customers to leave because of response times
    - We want as many customers as possible
    - The more auctions at a given time the lower we can set the price and the more customers we can get

Discuss what could be bottlenecks in your application and when they can appear and why?

Bottlenecks - Description

- The slowest component might become a bottleneck
- I/Os from the database
    - Could be fixed by caching often requested data

Make a data flow graph for your application and try to analyze it from data flow testing perspective. What is possible to see and what you cannot see in this test?

- Data flow graph
    - Graph where nodes are functions
    - Correct setting of session/cookie data
    - Find bottlenecks

The ebay case does not have any code implementation, we have therefore decided to create a data flow graph based on the following pseudo code:

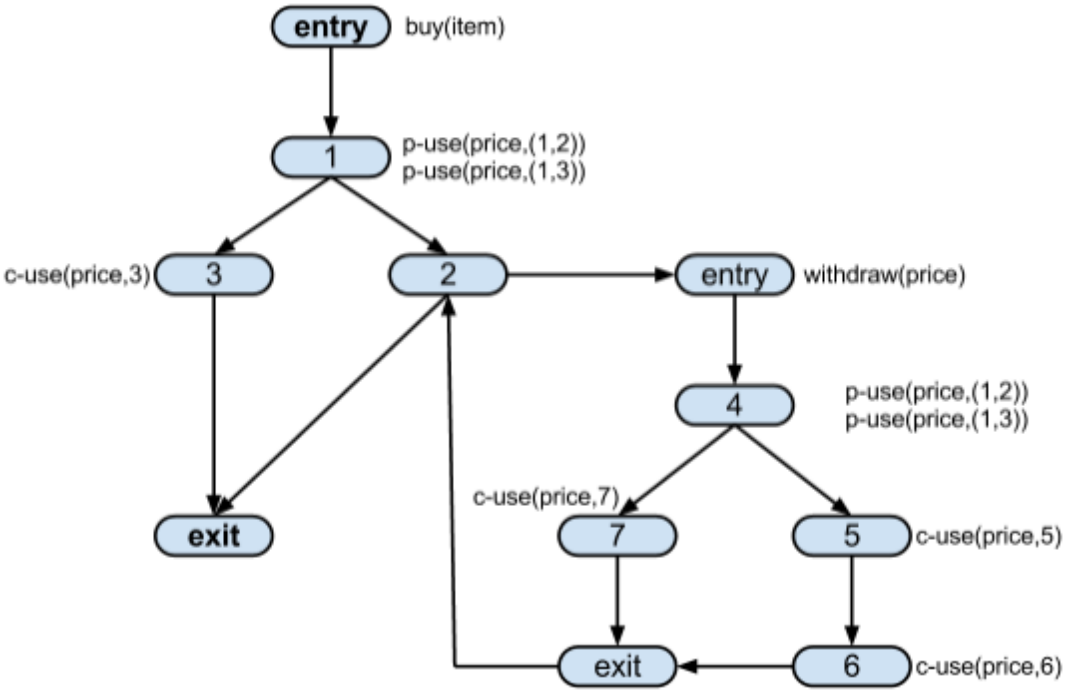
```
//Simple ebay buy function
//Input: item: The item you want to buy
  buy(item){
1.      if item is in stock
2.              withdraw(item price)
        else
3.              print: The item is not in stock.

//Simple ebay function to withdraw money from your account.
//Input: price: The price of the item you wish to purchase.
  function withdraw(price)
4.      if price >= money on your account
```

5. account money = (account money - price)
6. print: Item bought

else

7. print: you cannot afford the item



What are advantages of this test and disadvantages?

- Pros
  - Detect bottlenecks
  - Correct use of cookies and sessions
- Cons
  - Demands insight in the code
  - Takes long time to perform

How these tests are related to the standard for the terminology of testing?

- Similar to usual testing (TOV)
- White, Black, Gray box testing

8. Quality II: model driven, statistical, and usage testing

Based on your design, and requirements discussed in previous assignments, discuss how the design models can be utilized in testing of your application, how usage analysis can be taken into account, and how you can perform a statistical testing.

USAGE ANALYSIS:

- Get all url's accessed in one session by users
- rank them according to number of occurrences

pros

- good for simple static websites

cons:

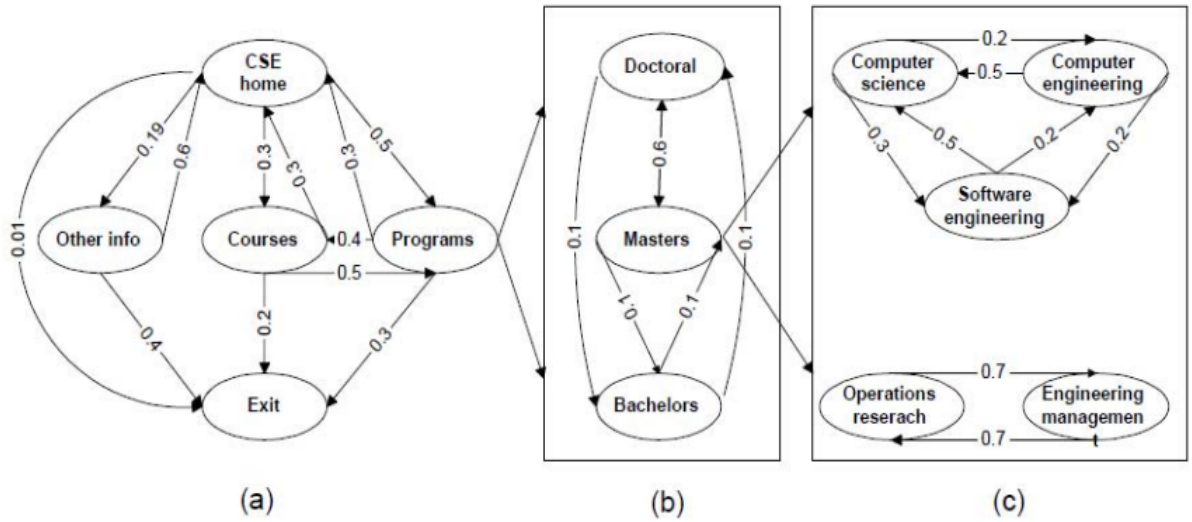
- complex application needs more fine grained log

used for:

- You want to typically find out whether your designed links are followed or not
- whether your pages are reachable
- which of the pages are not used and why
- which data items are used
- are there any anomalies, i.e. invoking functions after strange navigation sequence and so on

STATISTICAL TEST:

- Generation of test cases from usage data
- Targeting high risk areas
- Focuses on a probability distribution from application domain



pros

- Test cases generated randomly
- Amount of resources are reduced

cons

- Does not test unused features
- Cannot be used before deployment

Usual predictions focus on:

- Reliability
- Time to failure
- Mean time between failures
- All are of probabilistic nature

Reflect on advantages and disadvantages.
Look above.

In short, unused stuff does not get tested.

Answer how it would be possible to implement usage analysis extensions with log analysis library such as log4x where x is j, r, net, or php.

- Store the pages being requested, for later analysis
    - For later usage analysis and statistical testing

How these tests are related to the standard for the terminology of testing?
It is black box testing

- Similar to usual testing (TOV)
- White, Black, Gray box testing

9. Web Application Security

Take the design of your application in the context given not only by your current implementation but also by other exercises (especially navigation design and requirements).

Discuss which attacks are possible on your application and which techniques would you select for prevention and why.

Attacks

Input Validation Attacks -- Change the input after it has been validated
Form Tampering -- Buy/Bid form changed to accept invalid sums
Cross-Site Scripting (SSX) -- User submitted content can include SSX
SQL Injection -- To get credit card information on users
DDoS -- No users = no income
Privilege Escalation -- Gain more access by tricking the site to think its an administrator
Remote Malicious File Inclusion -- Put a file on a website
Canonicalisation Attacks -- Alternate references to the same path

Defence

Input type checking – Check for example for numbers
Encoding of inputs – Disallow special characters or special encoding schemes
Positive pattern matching – Recognize good input opposed to bad input
Identification of all input sources
String length – Restrict to the length of the field in the DB
String character distribution – Strings are often human readable, almost always contain printable characters

Show them also in the context of your implementation.

SQL injection attack:

```
String input = "harddrive; DROP TABLE Products; --";

Sanitize(input); // Removes illegal characters

query("SELECT * FROM Products WHERE ProductName LIKE " + input);
```

Cross-Site Scripting(XSS):

```
String input: "<script> redirect to maliciousssite.com </script>";

Sanitize(input);
```

```
query("INSERT INTO Comments VALUES " + input);
```

Form Tampering:
Change POST data in payment.

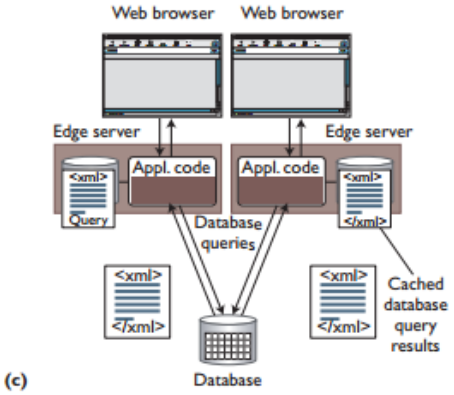
Before:
Price: 1000

After:
Price: 1

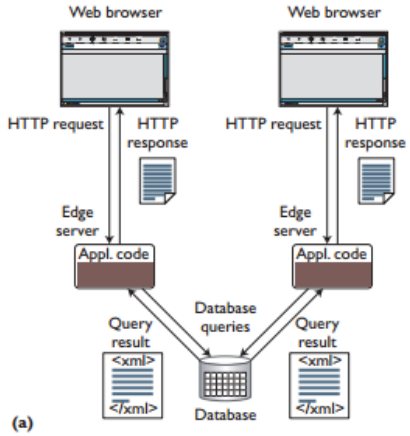
10. Scalability

Take your assignment application or your student project. Discuss what scalability strategy would be the best one for your application and why.

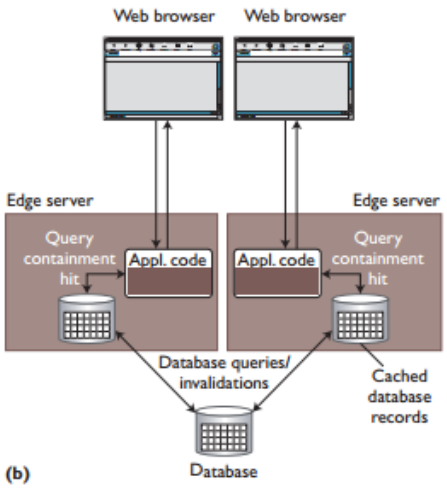
- Content Blind Caching



  - Store queries and results at edge server
  - Many similar queries on popular listings -- Tag Cloud
  - Won't query the often used queries at the DB because they are cached
- Edge Computing

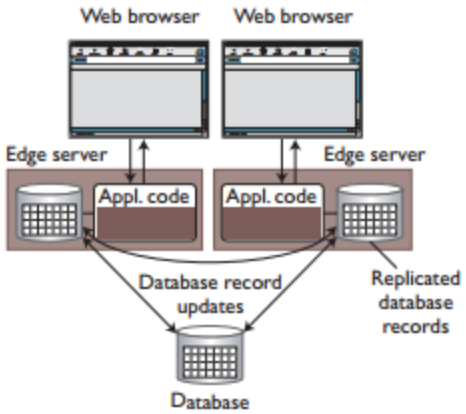


  - Application code is at several edge servers
  - Centralized server only handles user specific content
  - Will become a bottleneck when many users request at the same time
  - Good when most content is static at the edge servers
- Content Aware Caching



- Edge servers have a partial DB with previous query results
- Check if a query exist at the edge server before asking the central
- Can be used to evaluate queries executed by the search bar (min/max price) because they are very similar

- Data Replication



- Complete replica of the DB at the edge servers
- Have to update edge whenever the central is updated
- Much traffic when bidding on listings

Take at least one of the presented techniques and realize it.

```
public class CachedQuery{
      private Timestamp time;
      private String query;
      private String result;

      public static String Query(query){
            result = getResultFromCache(query);

            if(result == null){
                  result = getResultFromDb(query);
```

```
                addToCache(query, result, time.now);
            }

            return result;
        }
}
```

Discuss its advantages and disadvantages and its impact.

Advantages:

- Popular tasks are performed faster
- Overall load time reduced
- No duplicate queries at the database

Disadvantages:

- Unpopular tasks take longer
- Cache has to be cleared or updated if information is changed in the database (e.g. change price of item).

11. Service Based Integration

Present your implementation of web services interface for your integration with XML file load.

Solution 1

localhost:8080/parser/parse

```
Switch(extension) {
        case xml:
                parseXML(); break;
        case json:
                parseJSON(); break;
        default: return “Unknown file type”; break;
```

Solution 2

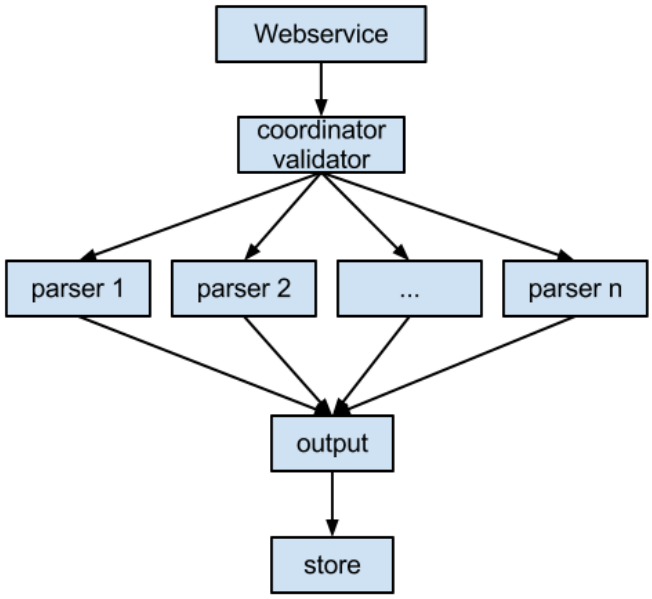
localhost:8080/parser/xml
        parseXML();

localhost:8080/parser/json
        parseJSON();

Design an infrastructure which would allow flexible addition of more not only XML resources.

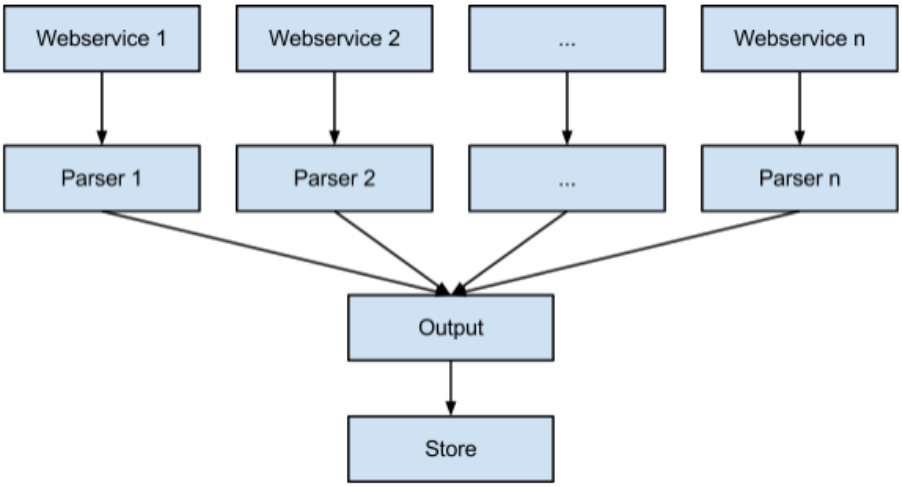
Solution 1

Single input that analyses the file and distribute it to the parser able to parse it to a unified output, that is stored and shown to the user.



Solution 2

Multiple input with a webservice for each filetype.



Discuss advantages and disadvantages of the solutions.

Solution 1

Advantages:

- Easy for the customer to find the entry point
- Only the coordinator have to scan for malware and SQL injection
- Only one webservice, it is therefore easy to maintain
- Code redundancy is low

Disadvantages:

- The singular webservice is a bottleneck, it limits to number of requests the service can handle
- The coordinator need to handle a lot of different types of file types.

Solution 2

Advantage:

- The multiple webservices can handle a lot of requests.

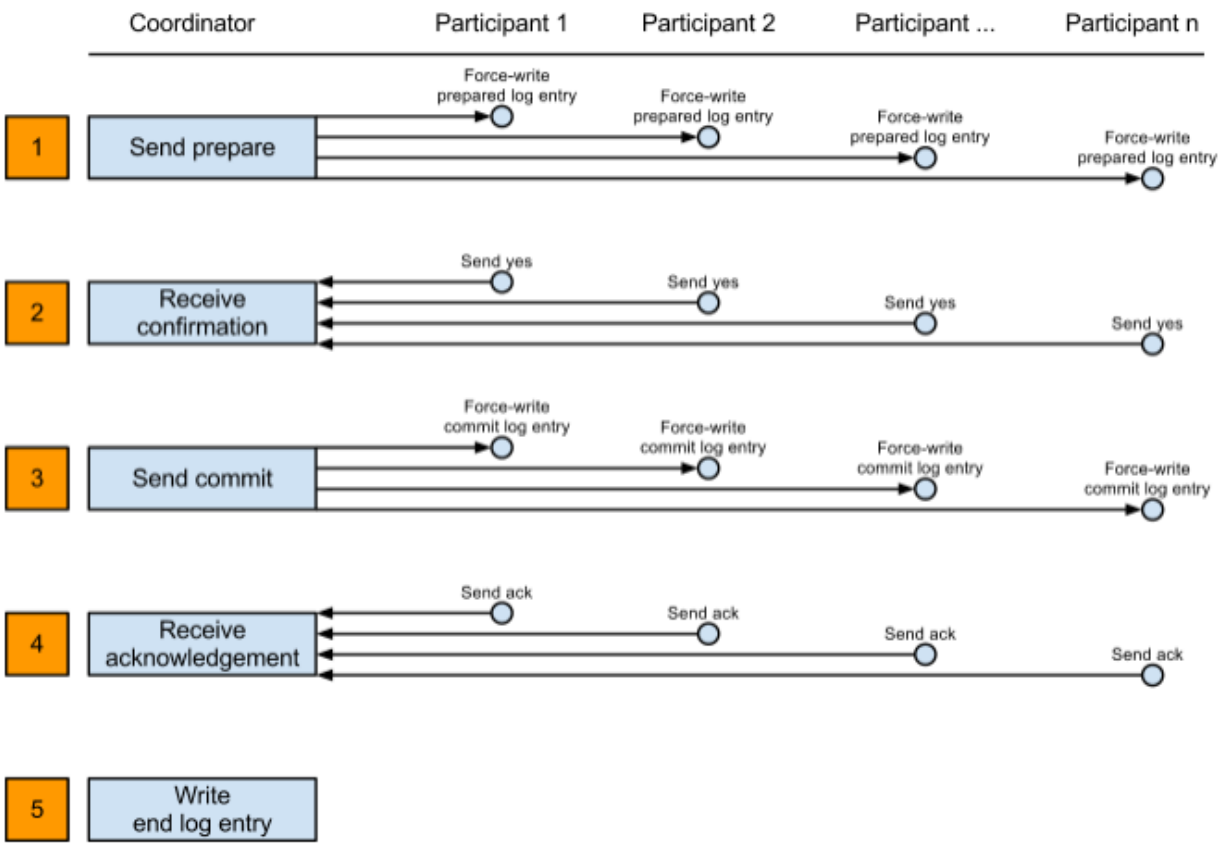
Disadvantage:

- Code redundancy is high
- Multiple webservices, which all need to be maintained.

12. Fault Tolerance and Transactions

Present your implementation of 2 phase commit protocol for your application.
Two phase commit protocol for a webservice running on distributed servers
The coordinator ensures a unanimous outcome of the transaction. Either all participants perform local commits or all perform local rollbacks.

1. Voting/Preparation phase - The coordinator ask all the participants if they are ready for a commit.
2. Coordinator receive answer from the participants. If all the participants reply “yes” the protocol continue to phase 3. If at least one of the participants either doesn’t reply “yes” or does not reply at all, the commit will stop.
3. All the participants is ready to receive a commit. The coordinator sends the commit to all the participants.
4. If the commit has been successful the participant will send an acknowledgement package back to the coordinator. If the coordinator does not receive an acknowledgement package from at least one participants, the commit is cancelled and the participants is required to perform a local rollback.
5. The commit is successful, and the commit is now finished.



Discuss advantages, disadvantages and reflect on problems.

Pros

- Distributed servers (Atomicity, Consistency, Isolation, Durability (ACID))

Cons

- Blocking protocol - Participants will never resolve their transactions if the coordinator fails permanently.
- A participant will be blocked after it has sent the coordinator an agreement message “yes”. The participant will be unblocked when a commit or rollback is received.
- Read locks
- Delay - 4 messages
- Availability - It is not certain that the webservice is always up due to blocks.

Problems

The major problem in distributed systems is that it can fail partially, so one or more servers fails while the others continue their normal operations. The servers exchange messages to agree upon whether a distributed transaction should be committed or aborted. Some of these messages might be lost due to local failures. This problem is solved by implementing a coordinator which makes sure that all the servers only commit, rollback, and abort if it is agreed upon. This solution has some advantages and disadvantages which can be read above.

What are optimizations you can make in your case if the number of applications grows.

Two Phase Commit optimization.