Testing Web-based Application

Ye Wu

http://www.ise.gmu.edu/~wuye

Web vs. Traditional Client-Server Systems

- Client-side application
  – GUI
  – Compatibility
- Server-side application
  – Control modeling
  – Compatibility
  – Web-specific features
  – Other issues, such as security and etc.

What to test

- User Interface Tests
- Functional Tests
- Database Tests
- Performance, load and stress tests
- Web security tests
- Configuration and compatibility tests
- Other tests

User Interface Tests

- User interface design testing
  – Profiling the target user
  – Data interaction (Data Input)
  – Data presentation (Data Output)

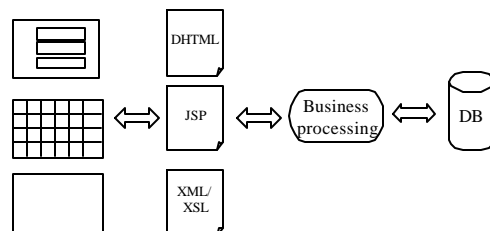- User interface implementation testing

---

User Interaction

- User interface control
  Graphic objects that enable users to interact with applications.
- Dynamic user interface controls
  – Javascript
  – Java
  – ActiveX
  – Server-Side Includes(SSI)
  – Style Sheet
- Navigation methods
- Feedback and error messages

Data Presentation

User Interface Implementation Testing

Capture/Replay tool

A capture replay tool is a set of software programs that capture user inputs and stores it into a format(script) suitable to be used at a later time to replay the user inputs.

Pros: Easy, fast, efficient.

Cons: When the GUI changes, input sequences previously recorded may no longer be valid.

Functional Tests

- Unit testing

- Integration testing
  - Black-box
  - White-box

Integration Testing

Atomic section is a static HTML file or a HTML section of a server program, which has an all-or-nothing property, i.e. either all the section is sent to clients or none of the section will be sent to clients.

```
PrintWriter out = response.getWriter();
out.println("<HTML>")
out.println("<Head><Title>" + title + "</Title></Head>)"
out.println("<Body>")
for(int I=0; I<myVector.size(); I++)
  if (myVector.elementAt(i).size > 10)
    out.println("<p><B>" + myVector.elementAt(i) + "</B></p>");
  else
    out.println("<p>" + myVector.elementAt(i) + "</p>");
out.println("</body></html>");
out.close();
```

$p1 = $ { out.println("<HTML>") out.println("<Head><Title>" + title + "</Title></Head>)" out.println("<Body>") }

$p2 = $ out.println("<p><B>" + myVector.elementAt(i) + "</B></p>");

$p3 = $ out.println("<p>" + myVector.elementAt(i) + "</p>");

$p4 = $ out.println("</body></html>");

Composite Section

p is a *composite section* of a server program P if
- p is an atomic section, or
- p_1 and p_2 are two composite sections, $p \circledR p_1 \cdot p_2$ is a new composite section, which servers will pass composite section p_1 followed by composite section p_2 to client. or
- p_1 and p_2 are two composite sections, $p \circledR p_1 | p_2$ is a new composite section, which servers choose to pass either composite section p_1 or p_2 to client, but not both at one iteration.

Interactions

- Link transition
  An invocation of a link in p causes the transition of section q from server to client.
- Operational transition
  The transition from p to q is not due to the link transition, but instead, is caused by the operations of client-side users, such as hitting a back button or a refresh button, which will load the previous pages, or reload the current page.
- Include transition
  Composite section p will include composite section q
- Forward transition
  Composite section p will forward the control to composite section q

```
<HTML><HEAD><TITLE>Grade Query Page</TITLE></HEAD>
<BODY>
<FORM METHOD=GET ACTION = "GRADE_SERVLET">
PLEASE INPUT YOUR ID AND PASSWORD:
<INPUT TYPE = "TEXT" NAME = ID SIZE = 10>
<INPUT TYPE = "PASSWORD" NAME = PASSWD SIZE=20>
<INPUT TYPE = "SUBMIT" NAME = SUBMIT VALUE=SUBMIT>
<INPUT TYPE = "RESET" VALUE = RESET>
</FORM></BODY></HTML>
```

```
p1  =   PrintWriter out = response.getWriter();
        out.println("<HTML>")
        out.println("<Head><Title>" + title + "</Title></Head>)"
        out.println("<Body>")
        if(VALIDATE(ID,PASSWD))
        for(int I=0; I< NUMBEROFCOURSE; I++)
p2=       out.println("<p><B>" + COURSE_NAME(I) + "</B>" + COURSE_GRADE (I) + "</p>");
        Else
p3  =   out.println("WRONG ID OR WRONG PASSWORD");
        out.println("<FORM METHOD=GET ACTION = "SEND_EMAIL">);
        out.println("<INPUT TYPE ="TEXT" NAME = "SUBJECT" SIZE = 50");
        out.println("<INPUT TYPE ="TEXTAREA" NAME = "BODY" WIDTH = 50");
        out.println("<INPUT TYPE = "SUBMIT" NAME = SUBMIT VALUE=SUBMIT>");
        out.println("<INPUT TYPE = "RESET" VALUE = RESET></FORM>");
p4  =   out.println("<A HREF =index.html> Go Back To Main Page </A>");
        out.println("</body></html>");
        out.close();
```

$S = \{ \text{index.html} \}$

$C = \{ \text{GRADE\_SERVLET} = p_1 \cdot (p_2^* \mid p_3) \cdot p_4,$

$\quad \text{SEND\_EMAIL} = ... \}$

$T = \{ S ==> \text{GRADE\_SERVLET},$

$\quad \text{GRADE\_SERVLET}.p3 ==> \text{SEND\_EMAIL},$

$\quad \text{GRADE\textbackslash\_SERVLET} .p3 ==> \text{index.html} \}$

Functional Tests

- Atomic section coverage
- Transition coverage

- Testing stateless application
- Testing stateful application
  Data-flow testing

Database Tests

- Test database design
- Test data content
- Test data integrity

Performance, load and stress tests

- Will the system be able to handle increases in Web traffic without compromising system response time, security, reliability, and accuracy?
- At what point will the performance degrade, and which component will be responsible for the degradation?
- What impact will performance degradation have on company sales and technical support costs?

Performance, load and stress tests

If a e-commerce site handles 300,000 transactions per day, so transaction response time = 300,00/[24*60*60] = 3.47

- If the transaction response time is greater than 4 but less than 9 seconds, 30% of users cancel their transactions.
- If the transaction response time is greater than 8 but less than 10 seconds, 60% of users cancel their transactions.
- If the transaction response time is greater than 4 but less than 10 seconds, 90% of users cancel their transactions.
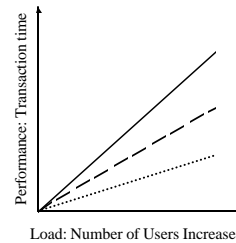
If the number of transactions is expected to rise between 25%-75%, and potential revenue for each transaction is $1, what are the revenue loses?

S = Server response time
N = Network service time
B = Browser processing time

Work load

Work load is the amount of processing and traffic management that is demanded of a system
– Users
– The applications
– Resources

Response and Performance Terms

- Throughput – The amount of data transmitted during client-server interactions.

For example, 10,000 concurrent users who request documents from a pool of 10 different HTML document(2k) every 3.5 minutes.

$$Throughput = \frac{10,000 \cdot (2 \cdot 1024 \cdot 8)}{(3.5 \cdot 60)} = 780,190 bps$$

Performance, load and stress tests

- Defining baseline configuration and performance requirements
- Determining the workload
- Determining when test should begin
- Developing test cases
- Analyzing and reporting collected data

Web security concerns

Vulnerabilities
– Software bugs
– Programming language bugs
  Javascript, java
– Active X
– Cookies
– Virus and worm
– Denial-of-service attacks
– others

Configuration and Compatibility Tests (1)

The objective of configuration and compatibility testing is to find errors in the application while it operate under the major real-world user environments.

Configuration and Compatibility Tests (2)

Server-side:
- – Application servers
- – Web servers
- – Databases servers
- – Firewall
- – OS
- – Hardware
- – Concurrent applications

Configuration and Compatibility Tests (3)

Client-side:
- – Brower type and version
- – OS
- – Firewall
- – Concurrent applications(instance messaging virus checkers, etc)
- – Client-side hardware such as printer, video and storage
- – Transmission control protocol
  - AOL, Microsoft networking, other third-party TCP/IP stack

Configuration and Compatibility Tests (4)

- Network devices and connectivity
  - – Bridge, routers, gateways, and so forth
  - – Internet/Intranet
  - – 10/100 base-t, modems, T1, ISDN, DSL, and so forth

Configuration and Compatibility Tests (5)

- Analyze market share
- Analyze the software on both the client side and the server side.
- Analyze the ways in which the server generates contents and in which the browser interprets, then formats and displays the contents
- Share experience
- Tool support