Detailed Design (Test Harness)

The test harness is described in the following sections.

Test Harness

Test Harness Setup

The test harness will be initially called from a starting web page that sends the name - value pair "testID=start" to the testHarness.cgi script.

Test Harness Execution

testHarness.cgi?testID=start

```
Read the identifying and list cookies
If identity cookie is undefined, then
        Create randomized list of interactors
        Create list of XHTML documents to send out
        Set sessionId
        Set identity cookie to sessionId
        Set list cookie to list of XHTML documents
        Send out cookies and first interactor test XHTML document
Else
        If requesting XHTML document contains interactors, then
                parse out the performance metrics
                insert performance metrics into database
                set list cookie to list cookie minus first XHTML document
                send list cookie and XHTML document
            Else, if  requesting XHTML document contains preference metrics
                    parse out the preference metrics
                update database with preference metrics
                set list cookie to list cookie minus first XHTML document
                    send list cookie and XHTML document
            Else
                    parse out the comments
                insert comments for interactors into database
                set list cookie identity and list cookies to null
                send cookies
```

Description

The test harness is a PERL cgi script that presents the test XHTML documents to the user and collects the results in an SQL database.

More specifically, the test harness creates a randomized list of interacor tests to be performed. Each interactor test consists of two XHTML documents. The first XHTML document contains the interactor to be tested and returns the performance metrics. The second XHTML document contains and returns the preference metrics. The last XHTML document collects and returns comments relating to all the interactors considered

together. After sending out the XHTML documents, the test harness collects the returned metrics or comments and stores then into the database, before sending out the next XHTML document.

The test harness also uses two cookies for determining which XHTML documents to send out next and saving the metrics and comments in related tables.

Cookies
The test harness uses two cookies, the identity cookie and a list cookie, as follows.

The identity cookie contains the sessionId, which denotes the test session, 1 through 40.

The list cookie contains a list of file names to be sent in sequence. The head file name denotes the current file sent to the client. The list of file names is made up of interactor htm and preference htm pairs, except for the last file name. The last file name denotes the interactor comments htm file that is sent as the last file of the test session. Although the files will be sent sequentially, the interactor and preference metrics htm file pairs are set in random order with respect to each other.

Test Harness Global Variables
The test harness sends out XHTML documents, which in turn requests the test harness. When the XHTML documents request the test harness, they will return name-value pairs to the test harness, as inputs. For each test session there will be three different types of name-value pairs, performance metrics, preference metrics, and comments about the interactors. As a result, the test harness will use the following variable names to hold the corresponding values.

For performance metrics, we have:

- testID: identifies test interactor by integer, values 1 - 12
- elapsedTime: number of milliseconds that the interactor runs
- match: number of matches where the user response is in the grammar
- noMatch: number of no match errors - user response not in the grammar
- noResponse: number of no responses
- nulls: number of nulls returned by the interactor
- correct: number or correct responses
- incorrect: number of incorrect responses
- missing: number or missing responses
- extra: number of extra responses
- error: error message
- response: string denoting the user's responses to interactor prompts
- secondLevelNo: number of user responses indicating "no" to 2nd level prompts
- topChoice: number of times first option is chosen

For preference metrics, we have:

- testID: identifies test interactor by integer, values 1 - 12
- clarity: 1-5 rating denoting user's understanding of what to do

- effectiveness: 1-5 rating denoting user's feeling about being able to complete task
- easeOfUse: 1-5 rating denoting how easy interactor was to use
- promptSpeed: 1, 3, 5 rating denoting speed too slow, OK, too fast
- multSelectEase: 1-5 rating denoting ease of selecting mult. options
- confirmation: 1-5 rating denoting preference for confirming choices at end
- deselection: 1-5 rating denoting how well deselecting, at end, worked
- secondPrompt: 1-5 rating denoting helpfulness of second prompt
- confirmCorrect: 1-5 rating denoting how well confirming/correcting choices at end worked

For comments concerning all the presented interactors, we have:

- bestLikedFeatures: string denoting what user liked about the interactors
- leastLikedFeatures: string denoting what the user least like about the interactors
- desiredFeaturues: string denoting what features user would like added
- recommends: string denoting whether user would recommend interactors to a friend
- comments: string denoting user comments or suggestions

Database Tables

The test harness will be adding and updating records in the interactorMetrics and interactorsComments tables.

Test Harness Functions

```
function randomInteractors() # returns list of interactors in random order

# returns list of XHTML documents to send out
function documentList(randomInteractors: list)

function sessionId()    # returns the session id number 1-40

# returns hash with name-value pairs as key-value pairs, denoting performance
# metrics.
function parsedPerformance()

# inserts performance metrics into interactorMetrics table
function insertPerformance(performanceMetrics: hash)

# returns hash where name-value pairs are key-value pairs, denoting prefercence
# metrics.
function parsedPreference()

# updates interactorMetrics table with preference metrics
function addPreference(preferenceMetrics: hash)

# returns hash, where name-value pairs are key-value pairs, denoting comments
```

about interactors.
function parsedComments()

inserts comments about interactors into the interactorsComments table
function insertComments(comments: hash)