Principles of Information and Database Management
198:336
Week 10 – Apr 11
Matthew Stone

Project Update

Where you should be
What to do next
Timeline for the rest of the semester

Designing an application

Example I've been working on
- Run survey experiment over the web

Background

Studying face-to-face communication
- We want to design animated characters that signal relationships and emotion like people
- We need to know what signals people use
- We need to know how to animate those signals

Methodology

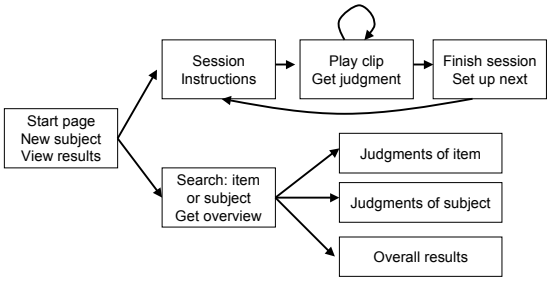
Four steps
- Collect and analyze recordings of people talking in conversation
- Ask other people what speakers seem to be doing – find reliable signals
- Develop a data format for specifying those signals in animation – XML!
- Ask about what animated characters seem to be doing – make sure signals are reliable

Web interface for step 2

Need an interface to
- Set subjects up for the experiment
- Show subjects data and collect their judgments
- View individual responses
- Analyze overall results for experiment

More Specifically

Map of pages in the interface



What this does

Organizes the code you have to write

- Each interface state corresponds to a page
- Have to write code for each page
  - In Java, one servlet class per page
  - Perhaps share low-level classes across pages

Highlights need for state in interface

Aside: State and the Web

HTTP has no state

- New connection for each request
- Browser sends all available information as part of the request
- Through parameters: get/post form inputs
- Through cookies: special attribute-value pairs

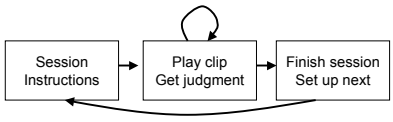
Aside: State and the Web

You have to be explicit about state

- Build suitable HTML on the fly
- Include "hidden inputs"
  <input type="hidden" name="x" value="y" />
- Specify actions in forms based on context
- If you're fancy, set cookies and get cookies

Map and State

Here, you need to keep track of key info:



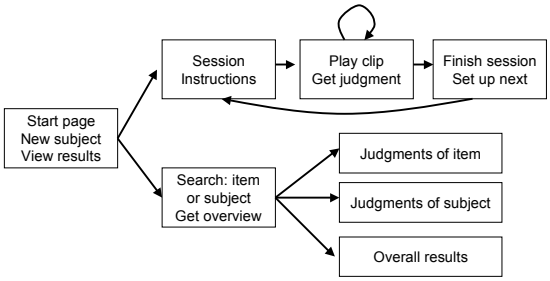
- User, Session type, Item list, Current item
- E.G. through hidden inputs

Fleshing out design

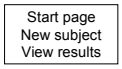
Spell out how pages interact with DB

- To take updates from last response
- To satisfy request
- To create page

More Specifically



Start screen

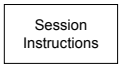


Start screen

Nothing has been input

No queries needed to make the page

Instructions



Instructions

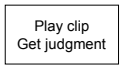
If “New Subject” action has just happened

- Must create a new ID for current subject

To create the page

- Must access the next protocol for this subject
- Update the DB to store this subject & protocol
- Save the ID & protocol & start info in the state
- Must get and display protocol instructions

Play clip, get judgment



Play clip, get judgment

If just got a judgment
- Insert new entry,
  based on user, session, protocol, clip, value

Get the next judgment
- Based on session and protocol
- Format page to play appropriate media
- Set up action to do based on what's left

Finish up session

Finish session
Set up next

Finish up session

If just got a judgment
- Insert new entry,
  based on user, session, protocol, clip, value
- Better reuse this code from clip page!

Determine what's next
- If there's another session, get ready to start
- Otherwise thanks for playing!

Experimenter's Search Interface

Search: item
or subject
Get overview

Experimenter's search interface

Always the same page
- Menus for kind of search
- Text field for search key
- Action for overview results page

Item Judgment

Judgments of item

Item judgment

Get results from database
- Based on search of judgments on this item

Format results as a table

Subject Judgments

Judgments of subject

Overall results

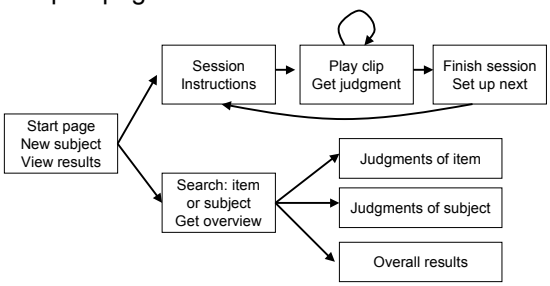
Overall results

Overall results

Get and display results
- Find conditions for experiment
- Select out the averages
- Display the results as a table (or graph, perhaps with error bars)

So there you have it

Map of pages in the interface



Really one third of the project!

Now you know exactly what DB stores
- Description of experiment
  protocols
  conditions
- Descriptions of items
- Judgments from subjects about items in conditions

Really one third of the project!

Now you know how to break up pages
- Know forms, links, queries you need
- Know what code can be shared across pages
- Know special structure on each page

Note on key features

A little of everything
- Updates as well as selects.
- More than one kind of user.
- Active links as well as forms.

Put yourself in your users' shoes
- Make something that fits them, their task

Next Step

Anyone can revise by Thursday 6pm.

Then, DB and SQL overview by recitation Wed 20.

DB and SQL

How will you store information?
- Relational schema for your stuff

How will you query it?
- For each page, what are the SQL commands
- Use ? notation for prepared statements
- Indicate how each of the ?s get values

What you should expect to do

Work from detailed map of your application
Write out the schema and queries on paper
Create the schemas in a real db
- E.G. oracle

Try out the queries with examples
- Make sure the results are what you expect
- If not, debug your queries!

What you should expect

By the time you hand this in
your project should be two thirds done!

Information Retrieval

Text as data
- legal decisions
- scholarly articles
- web pages!

Information retrieval

Relevance ranked query
- user specifies query terms
  words that are likely to occur in a document that they are interested in
- dbms returns an ordered list of documents
  documents higher in the list should match the query more closely than documents lower down

Vector space model

Text database with four records:

1 agent James Bond good agent
2 agent mobile computer
3 James Madison movie
4 James Bond movie

Just keep track of words that occur

Vector space model

New table:

document	agent	bond	computer	James	…
1	2	1	0	1	
2	1	0	1	0	
3	0	0	0	1	
4	0	1	0	1	

Dot product for similarity

Idea:
- two documents are similar if they have the same distributions of words
- intuition - they put the same emphases on the same concepts

Tweaking

IDF
- weight frequent words less

Length normalization
- weight longer documents less

Google PageRank – more next time
- weight indicative words higher
- weight “better” documents higher