conceptual models and conceptual design

by now (W07 pre-reading - Johnson)
- what conceptual models are (and are not), what they’re for, where they come from

upcoming workshop + assignments
- post-midterm, build conceptual model as part of early design

learning goals
- explain the purpose of a conceptual model and how it differs from a user’s mental model.
- explain the difference between a conceptual model and an interface design
- what are the risks and limitations of getting conceptual design wrong?
- list some of the components a conceptual model should include (e.g. metaphors, interaction types, objects/attributes, etc); given a scenario, identify examples of each;
- give some examples of way you could visualize a conceptual model

quiz results
- Q1: 48.6% said both (b) and (c), 39.3% said all of (a) (b) and (c)
- Q2: 96.3% correct
- Q3: 96.3% correct
- Q4 example answers:
  - Objects: review, rating, restaurant
  - Attributes: date, author, name, owner, location
what is conceptual design?

about crossing the gap from requirements to actually designing a solution

starts with brainstorming; multiple iterations to narrow down

a **conceptual model** is an outcome of conceptual design
- sometimes people will call this a ‘conceptual design’

a conceptual model
- can take many different forms
- be built through many approaches
- is essentially a set of ideas
mental models vs. conceptual models

**mental models:** *something the user has (forms)*
- users “see” the system through their own mental models
- users *rely* on mental models during usage
- there are various *forms* of mental models
- mental models can *support* or *impede* users’ interaction

**conceptual models:** *articulation of designer’s (i.e. your) mental model*
- what users will be able to do,
- how users should *feel* when using the product
- what *concepts or knowledge* users will need to interact
- how they will interact with it (at a very high-level)

---

**how does the conceptual model impact the interface design?**
creates a foundation for what needs to be in interface
> “If it isn’t in the conceptual model, the system should not require users to be aware of it.”

rare that a conceptual model is completely new
- common ways to do things will inform your model AND design
  - e.g., ‘shopping carts’

the conceptual model will impact the architecture of the system, which impacts what’s possible with interface designs.
> but you can always implement the SAME conceptual model with different looks and feels, different widgets, different layouts

---

**system design vs. interface design?**

system designers and implementers may have more concepts or details going on in the background

but conceptual model (and eventually interface) should only contain what users need

system concepts should only be included when they can foster a **good mental model**
for a conceptual model to work…

- It must make sense
e.g. metaphors that build on something the user knows, and translates well
- Focus on elements of task user wants to do
- It has to be consistent
e.g. in terminology, in how objects are interacted with, etc.
- Concepts are minimal
keep it simple as possible;
- Need to settle on it EARLY in the process

components of conceptual models

- Design metaphors and analogies
- Concepts – objects, attributes, operations
- Relationships among concepts
- Mappings from concepts to the actual task(s)
- Terminology that will be used (consistently) to tie it all together
- Interaction types
- Interface types

metaphors

Well known concepts you can rely on to help users understand and interact with the system

- Many kinds, e.g.
  - Interactions
    - Swipe to turn page in an ebook
    - Move backwards through time to explore file backups
  - Ecological, contextual, broader system structure, e.g.
    - Dropbox: a box you drop everything into
    - iCloud: central mother ship to which everything connects
  - Emotional, e.g.
    - Siri as a personal assistant
example: the desktop metaphor

unifying set of concepts employed in graphical user interfaces to help users understand and easily interact with a computer

computer monitor → user's desktop

objects → documents, folders

you can do things with these objects:

• place documents upon desktop
• open documents into a window → paper copy
• organize in folders

extend desktop with desk accessories → calculator, notepad

seems pretty obvious now, doesn’t it?

1970: from Alan Kay - xerox parc

next 10 years: innovated at PARC

1981: Xerox Star = 1st commercial system

At that time, with a computer you could:

• edit a text document, and print it
• do simple graphics ("paint")
• compute using something like a spreadsheet
• mouse for input (not just a keyboard); windowing systems

But the 'personal computer' didn’t exist.

Most people had never used a computer. If they had, it was a "command line" interface, usually shared (mainframe).

→ unify these operations into something comprehensible.

choosing metaphors

1. Identify the functional requirements
2. Understand which bit of the product are likely to cause users problems
3. Generate metaphors. Look for metaphors in the users’ description of tasks, as a starting point, or metaphors already used in the application domain.
identifying concepts: object / operation analysis

method from Johnson reading

⇒ what are all the ‘concepts’ that a user will need in the system?
⇒ implication: should be what people use, interact with in the interface!

INCLUDE: all objects, attributes, operations of tasks that users need to be aware of or understand to use system

• user-understandable entity types (objects, people, …?)
• attributes of each entity-type
• operations that users can perform on each type of object

⇒ note where these concepts may be different for different users!
⇒ note that there may be more than one way to organize this!

task examples are a great resource for these!

activity 1 - work through example: personal mp3 player

what concepts does the user need to know/see?

<table>
<thead>
<tr>
<th>objects</th>
<th>attributes</th>
<th>operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>songs</td>
<td>album, title, artist, description, currently playing, # times played, date added to system</td>
<td>play, preview, pause, stop, rewind, fast forward, add to playlist, send to a friend</td>
</tr>
<tr>
<td>album</td>
<td>title, artist, description, compilation, currently playing, # times played, date added to system</td>
<td>play, stop, add to playlist, send to a friend</td>
</tr>
<tr>
<td>playlist</td>
<td>title, description, date created, # times played</td>
<td>play, stop, skip song, choose song, send to a friend</td>
</tr>
<tr>
<td>user profile</td>
<td>username, favorite albums, favorite songs, credit card number</td>
<td>review songs, review albums</td>
</tr>
</tbody>
</table>

relationships

what actions/attributes are shared between objects?
- e.g., song, podcast, audiobook all have timelines that users want to navigate (i.e., fast forward, rewind, etc.)
- example interface design implication: make the interaction work the same for all, so user recognizes it.

containment and hierarchy
- e.g., a song is contained by an album
- interface design implication: represent this containment in the actual design of the interface
relationships cont. . .

how do objects, attributes, actions, etc. vary in importance?
- e.g., in managing a mp3 player, playing songs is a frequent task, while reviewing albums may be infrequent
- possible interface design implication: frequent tasks and actions should be easy to access; less frequent can be hidden if space is limited

mapping

how do the concepts map to what people will actually do?

i.e. mapping to a task, what users will actually do

one easy method:
"run" a task example on it

learn:
• are these the right objects? do they match what people actually have
• can i do all the operations? do they match what people want to do?
• can i do them in a consistent way?
• what will the experience be like? in what context will it take place?

terminology

what terms will you use to communicate concepts?

choose your terminology and stick to it!
easy to go from planning to interface and minimize confusion

does your user login to a system with a user-id? a username? a member id? or an email address?

interaction types vs. interface types

• interface type: what the thing being interacted with is
  • broad and loosely defined. e.g., command line (how you talk to it), intelligent (function), gestural (hardware), touch (both hardware and interaction type)
• interaction type: the nature of the thing the user is doing with it
  • should be dictated by the task, the users' abilities and needs
  • your textbook presents as four types:
    ➔ instructing, conversing, manipulating, exploring
  • but could define other ways!
    ➔ by context (e.g., problem-solving, learning, socializing, etc.)
    ➔ by activity (e.g., driving, at home, at work, etc.)
interaction types
instructing

user **issue instructions** to a system – telling it what to do
*RSP defines as *indirect* (as opposed to ‘direct manipulation’)*

common conceptual model:
- word processors (open, close, save, count the number of words, check spelling, etc.)
- VCR/DVD players (play, rewind, pause, etc.)
- etc.

benefit: supports quick and efficient operations
- good for repetitive actions on more than one object

interaction types
conversing

user has a **conversation** with a system… really a dialogue/conversation, not just a series of options and selections.
*human-like interaction*

more of a 2-way conversation than in instructing
- examples: usually implemented with natural language
  - many online help centers
  - Siri

benefit: when/WHY to use?
- good for novices, the computer phobic, specialized applications, etc.

interaction types
manipulating

use when:
- makes sense to **directly manipulate** of objects
- benefit: leverages what people do in the real world; (e.g., drag/drop)
- but CAN be used for non-realistic actions too (e.g., zoom)

- principles:
  - representation is always available (visible)
  - incremental, reversible actions (“undo”)
  - physical actions (drag/drop) rather than syntactic commands

- examples of tasks that could use “manipulating”
  - moving selected block of text around on a PowerPoint slide
  - touch interaction with maps (pinch, zoom, slide)
manipulating example


manipulating example

interaction types

exploring

user moves through virtual or physical environments

use when:
• user needs to explore and interact with an ‘environment’.
• can exploit user's previous knowledge of how they move through spaces (digital and physical)

examples of tasks that could use “exploring”
• finding a location in google maps: 
  using street view
• identify location using ‘dot’ on GPS: 
  physically move through actual environment with phone

interaction types

• instructing, manipulating: most common historically; but conversing and exploring increasingly used

• not exclusive
• you can do multiple within one interface for DIFFERENT objects
• or for the SAME objects, e.g., 
  ➔ instructing AND manipulating of files (open, close, save, etc.)
  ➔ instructing AND conversing for help functions
  ➔ conversing AND exploring for following GPS directions
interface types

huge range of possible interface types
- RSP Table 6.1 on pg. 158 for full list
- includes: data visualizations, command-based, WIMP and GUI, mobile, tangible, touch, etc.

➤ brainstorm broadly before narrowing

e.g. imagine you’re making a tool to teach people how to cook
- should it be built into a mobile phone? wearable? virtual reality?

<table>
<thead>
<tr>
<th>Interface type</th>
<th>See also</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Command-based</td>
<td>WIMP and web</td>
</tr>
<tr>
<td>2. WIMP and GUI</td>
<td>Augmented and mixed reality</td>
</tr>
<tr>
<td>3. Multimedia</td>
<td>Multimedia</td>
</tr>
<tr>
<td>4. Virtual reality</td>
<td>Mobile and multimedia</td>
</tr>
<tr>
<td>5. Information visualization</td>
<td>Mobile</td>
</tr>
<tr>
<td>6. Web</td>
<td>Augmented and mixed reality</td>
</tr>
<tr>
<td>7. Consumer electronics and appliances</td>
<td>Touch</td>
</tr>
<tr>
<td>8. Mobile</td>
<td>Speech, pen, touch, gesture, and haptic</td>
</tr>
<tr>
<td>9. Speech</td>
<td>Shareable, touch</td>
</tr>
<tr>
<td>10. Pen</td>
<td>Shareable, air-based gesture</td>
</tr>
<tr>
<td>11. Touch</td>
<td>tangible</td>
</tr>
<tr>
<td>12. Air-based gesture</td>
<td>multimodal</td>
</tr>
<tr>
<td>13. Haptic</td>
<td>Speech, pen, touch, gesture, and haptic</td>
</tr>
<tr>
<td>14. Multimodal</td>
<td>Touch</td>
</tr>
<tr>
<td>15. Shareable</td>
<td>Virtual reality</td>
</tr>
<tr>
<td>16. Tangible</td>
<td></td>
</tr>
<tr>
<td>17. Augmented and mixed reality</td>
<td></td>
</tr>
<tr>
<td>18. Wearable</td>
<td></td>
</tr>
<tr>
<td>19. Robotic</td>
<td></td>
</tr>
<tr>
<td>20. Brain-computer</td>
<td></td>
</tr>
</tbody>
</table>

a conceptual model excludes

- low-level presentation
- implementation details
- menu and screen designs
- widgets
- etc.

activity 2 - dissecting an interface design
what concepts did the designers start with?

StickIT. Mockup for a 'ubiquitous notes' mobile app.
- users post ‘sticky’ notes around UBC
- other users pick up notes; interact with notes

concept:
notes

attributes:
- titles
- categories
- dates
- content
- ratings

mapping:
navigate between notes

relationship:
notes are contained in a feed

interaction type:
exploratory

operations:
readers can rate notes, favorite notes

metaphor:
post-its notes to stick info to stuff

StickIT. 2014.
courtesy of Jessica Dawson, Juliette Link, Thea van Rossum
dissecting an interface design

what concepts did the designers start with?

StickIT. Mockup for a 'ubiquitous notes' app.

- users post 'sticky' notes around UBC
- other users pick up notes; interact with notes

StickIT. 2014.
courtesy of Jessica Dawson, Juliette Link, Thea van Rossum

concept:

- note

attributes:
- titles
- categories
- dates
- content
- ratings

mapping:

navigate between notes

relationship:
notes are contained in

metaphor:
post it notes to stick info to stuff

many other concepts at work behind this interface design!

StickIT. 2010.
courtesy of Jessica Dawson, Juliette Link, Thea van Rossum

based on those concepts

. . . StickIT interface could have also looked like this!


curtesy of Jessica Dawson, Juliette Link, Thea van Rossum

components of conceptual models

- any central design metaphors and analogies
e.g. the "desktop metaphor"
- concepts – objects, actions you can do to them, attributes
e.g., files and folders; both can be opened, have names;
- relationships among concepts
e.g., files are contained in folders
- mappings from concepts to the user experience envisioned;
e.g., the users can browse files, and mark favorites
- terminology that will be used (consistently) to tie it all together

- interaction types; how will they interact with it?
e.g. give commands, perform operations, explore
- interface types; what the thing being interacted with is

what a conceptual model looks like
what does a conceptual model look like

however best helps you describe and understand its components:

• written descriptions
• diagrams
• storyboards
• sketches
• lists and tables
• mood boards
• physical ‘sketches’

different methods might capture different parts of more effectively than others

➔ you’ll likely use a combination of more than one!

concepts, relations, terminology

one possible conceptual model representation for a music player:

• objects represented by boxes
• lines and labels indicate relations
• terminology = names given to objects

This conceptual model diagram does NOT show what you can do with it; It describes what system consists of and how it is organized.

CM for a debit machine

another one:

using a diagrammatic approach

shows concepts, relationships, terminology

metaphors for expressing emotion in a mobile phone

started as a list of words

Dawson, J.Q., Ferstay, J., Link, J., Haddad, S. Early brainstorming for the DEEVA project
storyboards and sketching

- flexible methods for representing conceptual design!
  - can be used to show what the user is thinking/feeling
  - communicate metaphors
  - interface types and styles of interaction
  - environments and contexts in which system is used

- can be very low investment

metaphors later became sketches

metaphors finally expressed as physical sketches

this storyboard illustrates an envisioned context and process
this sketch demonstrates an interface type and aspects of the interaction types (exploratory)

this conceptual design representation emphasizes objects and relationships for an e-ticket system

**storyboards and sketching**

you don’t need to be good at drawing to communicate your ideas

➤ sketches and storyboards can vary in fidelity

**storyboards**

guidelines for storyboards:

- decide what you want trying to communicate
- consider characters, plot, environment, user’s thought process and emotions
- iterate: start with text and arrows & move up to more and more involved drawings
storyboarding resources

- Ben Crothers. Storyboarding & UX – Parts 1 and 2
- Course textbook: pg 393 – 394

mood boards

conceptual models in 344 project
Brainstorm alternative design approaches to support your task examples.

Then develop and roughly sketch out at least two of your top ideas. For each of your top ideas, identify:

- the metaphor(s) or analogies the idea employs
- the concepts (including objects, attributes, operations)
- relationships among concepts; and mappings from concepts to the actual task
- interaction types and interface types

activity 3 - scenario

**Imagine:** you’ve been hired to (eventually) build a new user web interface for reserving student study rooms in the UBC CS department.

In this system, users must be able to:

- log on with their department ID
- see what rooms exist (list or map view)
- see and search room availability
- reserve a room (if it is available), and receive an email confirmation sent to their department ID
- hold one future room reservation at a time
- see their own future reservation, if any.

If a room has already been reserved by someone else, students should not be able to find out who has reserved it, but users with tech staff credentials should be able to find this information.

activity 3 - steps

1) Brainstorm visual representation(s) of a conceptual model
   ➔ using crazy 8s

2) Identify components of your concept model(s)
   ➔ brainstorm on each component that we’ve discussed

summary: a good conceptual model...
summary: conceptual model mistakes to avoid

- don’t include non-task relevant concepts
  e.g., don’t expose implementation if not necessarily

- don’t confuse concepts
  e.g., in terminology, in how objects are interacted with, etc.

- don’t use non-task relevant terminology
  e.g., pick a term and stick to it!