Using Conceptual Models in Interaction Design

June 6, 2012

Hugh Dubberly
Dubberly Design Office
When you say, “I know how to use Photoshop,” or “I know how to use Excel,” you refer to a conceptual model.

Knowing how to use an application—or any tool—means having a conceptual model of the application.
At their core, all applications have a main data type, and tools for selecting and changing data.

If you understand the main data type and the tools for selecting and changing data, then you understand the application—or at least much of what it can do.
What defines a spreadsheet?

Spreadsheets are **grids of cells**, containing **text, numbers, and functions** that operate on a range of cells.

This is what makes VisiCalc similar to Excel.
What defines an image editor?

Images are **grids of cells**, containing **numbers (that represent colors)**. You edit an image by selecting a range of cells and applying a transform function. This is what makes MacPaint similar to Photoshop.
I believe that understanding the conceptual model is key to interaction design;

that’s because users need to understand the model in order to be able to use a product easily.
Don Norman talks about a similar idea: The system image

“For people to use a product successfully, they must have the same mental model (the user’s model) as that of the designer (the designer’s model). But the designer only talks to the user via the product itself, so the entire communication must take place through the ‘system image’: the information conveyed by the physical product itself.”

In addition to communicating the conceptual model to users, designers also need to communicate it to other designers—and to product managers and senior management, and, of course, to engineers.

And the communication needs to go two ways; it needs to be a conversation.
The idea of a conversation about conceptual models of a product or service raises some questions.

How might we design these conversations? How might we make them rigorous?

What sorts of artifacts might support them? Is there a necessary and sufficient set of artifacts?
Foundation
Steve Job’s definition of design

“In most people’s vocabularies, design means veneer. It’s interior decorating. It’s the fabric of the curtains and the sofa. But to me, nothing could be further from the meaning of design. Design is the fundamental soul of a man-made creation that ends up expressing itself in successive outer layers of the product or service.”

—Steve Jobs, Fortune, January 24, 2000
“In most people’s vocabularies, design means veneer. It’s interior decorating. It’s the fabric of the curtains and the sofa. But to me, nothing could be further from the meaning of design. Design is the fundamental soul of a man-made creation that ends up expressing itself in successive outer layers of the product or service.”

—Steve Jobs, Fortune, January 24, 2000
Should we take Steve Jobs literally? Is it worth asking:

What’s a product’s fundamental soul?

What are the successive outer layers?
Let’s start with a model of the layers. Perhaps simplest: 2 layers

- Function

- Form

—Louis Sullivan, *The Tall Office Building Artistically Considered*, 1896
Linguistics and Semiotics (the theory of signs) suggest 3 layers

- **Pragmatic** or Context or Why are we doing this?

- **Semantic** or Meaning or What are we doing?

- **Syntactic** or Grammar or How are we doing it?

—Charles W. Morris, *Foundation of the Theory of Signs*, 1938
Looking at components may suggest 4 or even more layers

- **System**: a community or marketplace of products interacting
e.g., the Kindle, reader application, Whisper-net network, and Amazon store eco-system

- **Product**: a series of connected and related screens
e.g., the iPhone as a new paradigm in touch-screen-based mobile computing UIs

- **Screen**: a collection of interaction components
e.g., the Windows Phone 7 home screen

- **Widget**: an interaction component, often repeated
e.g., one-button check-out or fan-able pull-down menus
Planning roles may suggest 5 layers

- **Strategy**: User needs + site objectives

- **Scope**: Functional + content specs

- **Structure**: Information design (interface + navigation)

- **Skeleton**: Interaction design + Information architecture

- **Surface**: Visual design

And we could continue to divide more finely, or we could slice along different dimensions.

But the main point is that the layers depend on each other. And to a large extent—at least in great products—the successive outer layers derive from a core, from a fundamental soul.
Most UX design takes place at the outermost layers, at the screen level—by iterating on wireframes or Photoshop mock-ups.

*Iteration at the screen level is necessary, but sometimes a focus on details can hide the big picture; it can hide the fundamental soul of the product.*
Historically, design hasn’t paid much attention to concepts, to methods for defining the fundamental soul of a product.

We have a few: elevator pitch, story treatment, competitive positioning, value proposition, unique selling proposition, solution space models,—and now conceptual models.
Definitions
It may be easier to remember a simpler model, nicely parallel to the Charles W. Morris model

- Concepts
e.g., all concepts that the application’s user interface exposes to users

- Task flow
e.g., the sequence of operations that users execute to accomplish tasks

- Presentation
e.g., the controls, displays, etc. that comprise its user interface

“... interaction design consists of concepts, task flow, and presentation.”

—Jeff Johnson + Austin Henderson, Conceptual Models: Core to Good Design, 2012
From a wonderful new book
“A conceptual model is a high-level description of an application. It enumerates all concepts in the application that users can encounter, describes how those concepts relate to each other, and how those concept fit into tasks that users perform with the application.”

—Jeff Johnson + Austin Henderson, *Conceptual Models: Core to Good Design*, 2012
More precisely, a conceptual model of an interactive application includes, the data “objects,” which a user may encounter, and the “operations, attributes, and relationships,” which a user may perform on the data objects.
It’s important to note that the conceptual model is independent of task flow and presentation.

As you work out task flow, you may find problems in the conceptual model, and you may need to modify it.
As an example of a conceptual model, Johnson + Henderson describe an alarm clock.

The clock stores the current time of day, continually updating it to track the passage of time.

It displays the current time constantly.

Users can set the current time.

Users can set an alarm at a specified time, or no alarm.

When an alarm is set and the current time equals the set alarm time, the alarm is triggered.

Users can turn off an alarm.
Johnson + Henderson’s conceptual model of a clock could be expressed in digital or analog travel clocks.
Their conceptual model of a clock could also describe the iPhone and Galaxy Nexus clocks.
A critique and a proposal
Lists are not models.
Sentences are not efficient means of describing structure.
Text is often redundant.

Node-link diagrams—concept maps—are a more efficient means of representing conceptual models
A conceptual model of an alarm clock represented as a concept map.

Clock includes Current Time
  Continually Updates
    can be
    Displayed
    Set (Changed)

Clock includes Alarm Time
  can be
  Set (Changed)
  Enabled
  Disabled (Turned Off)
  Triggered
  (When Alarm Time ≥ Current Time)
Conceptual model of an alarm clock with detail added.

Clock

- includes
  - Current Time
    - Continually Updates
    - can be
      - Displayed
      - Set (Changed)
  - Alarm Time
    - can be
      - Set (Changed)
      - Enabled
      - Disabled (Turned Off)
      - Triggered
      - (When Alarm Time ≥ Current Time)
- includes
  - Hour
    - can be
      - 1-12 or 1-24
  - Minute
    - can be
      - 0-59
  - Second
    - can be
      - 0-59
  - Time Format
    - can be
      - Standard Time 12-hour, AM/PM
      - Military Time 24-hour
  - AM/PM
    - can be
      - AM, PM or Neither (if set to Military Time)
We might formalize the representation.

Black = data objects
Blue = relationships
Arrows = data structure
Green = operations
Tinted colors = future
Health challenge example

- **Accounts**:
  - can be created/updated/deleted
  - require email/password
  - have password reset

- **Email**:
  - can be sent/received/accepted/declined
  - sent to anyone

- **Wellspace Members**:
  - can send/received/accepted/declined

- **Profile Picture**:
  - can be created/viewed/updated

- **First Name**
  - Last Name
  - Profile Picture
  - Gender
  - Time Zone
  - Challenges won
  - Active challenges
  - Invisible
  - Visible

- **Apps**
  - can be built in/added

- **Apps**
  - can be viewed

- **Apps**
  - can be built in/added from the App Store

- **Data**
  - can be sent to devices

- **Devices**
  - can be linked/viewed/added/deleted

- **Register**
  - can have email/password
  - require email/password
  - have password reset

- **Email**
  - can have Gmail Contacts

- **Profile Picture**
  - can be displayed/edited

- **Profile Picture**
  - can be created/viewed

- **Notifications**
  - can be displayed/cleared

- **Photo**
  - can be sent to email

- **Profile Picture**
  - First Name
  - Last Name

- **Password**
  - can be forgotten/reset

- **Profile Picture**
  - can be added/deleted

- **Profile Picture**
  - can be created/viewed

- **Profile Picture**
  - can be viewed/deleted

- **Profile Picture**
  - can be added/deleted from searched

- **Challenge Invites**
  - can be sent/received/accepted/declined

- **Rankings**
  - can be top player/top team

- **Titles**
  - can be set/changed

- **Team**
  - can be added/deleted from

- **Wall Posts**
  - can be added/deleted

- **Winning Conditions**
  - can be set/changed

- **Value**
  - unit
  - type
  - period
  - - activity (steps)
  - - blood glucose
  - - blood pressure
  - - cholesterol
  - - food
  - - weight

- **Winning Conditions**
  - can compare to

- **Weekly**
  - can be daily/weekly/bi-monthly/monthly/yearly

- **Weekly**
  - can repeat

- **Weekly**
  - can measure

- **Weekly**
  - can trigger

- **Weekly**
  - can be ongoing/discrete

- **Weekly**
  - can be created/viewed/updated/compared

- **Weekly**
  - can be added/deleted

- **Weekly**
  - can be displayed/edited

- **Weekly**
  - can be set/changed

- **Weekly**
  - can compare to

- **Weekly**
  - can measure

- **Weekly**
  - can trigger

- **Weekly**
  - can be ongoing/discrete

- **Weekly**
  - can be created/viewed/updated/compared

- **Weekly**
  - can be added/deleted

- **Weekly**
  - can be displayed/edited

- **Weekly**
  - can be set/changed

- **Weekly**
  - can compare to

- **Weekly**
  - can measure

- **Weekly**
  - can trigger

- **Weekly**
  - can be ongoing/discrete

- **Weekly**
  - can be created/viewed/updated/compared

- **Weekly**
  - can be added/deleted

- **Weekly**
  - can be displayed/edited

- **Weekly**
  - can be set/changed

- **Weekly**
  - can compare to

- **Weekly**
  - can measure

- **Weekly**
  - can trigger
Changes after wireframe design
Case study
Orca: Blood Analyte Meter, early version

- **Meter**
  - **Screen**
  - **All results**
    - **Test result**
      - **Meal tags** (D, M, Y, H, M, S)
      - **Test time**
      - **BG value**
  - **Displayed as**
    - List
    - Individually
    - Graph

- **Strip**
  - **Channel**
  - **Inserted into**

- **User**
  - **Blood sample**
  - **Applied to**

- **Glucose (or other analyte)**
  - **Displayed on**

- **Slot**
  - **Inserted into**

- **Recorded as**
Orca: Blood Analyte Meter, revised version

Orca performs tests to produce results.

Results can be created, read, tagged, and stored (automatically after memory is full).

Date can be changed, date format can be set to mm/dd/yyyy, dd/mm/yyyy, yyyy/mm/dd.

Target ranges can be customized, enabled, disabled, with default values suggested.

Orca tests results can be created, read, tagged as deleted (automatically after memory is full), stored in glucose, HDL, cLDL, total cholesterol, triglycerides, etc.

Future versions may include 1,5 Ag, A1c, ALT/AST, BNP, creatinine, glycated albumin, ketones, white blood cell (WBC), INR, hs-CRP, or combination of above.

Test type, analyte(s), analyte value(s), meal tag, date + time, patient ID, device ID, no tag, before breakfast, before lunch, after breakfast, after lunch.

History can be viewed, filtered.

Communication abilities can be fit, linxx, patient (primary), physician, physician assistant, lab tech.

Language can be set to English (first version), Nederlands, Français, Deutsch.

Time can be changed, time format can be set to 12-hour, 24-hour.

Normal ranges can be customized.

Units per volume measured in mg/dL, mmol/L, etc.

Graphs can be measured in units per volume.

Settings include language, units per volume, graphing and tagging, patient ID.

Patient IDs can be turned off, graphing and tagging can be enabled, disabled.

Graphing and tagging can be turned on, patient IDs can be enabled, disabled.

Sound can be enabled, disabled.

Analyte, multiple results (values), units (mg/dL), data, meal tags, filters, goal range.

Analyte value(s) includes units (mg/dL), data, time, meal tags, filters.

Turn off graphing and tagging can be enabled, disabled.

Graphing and tagging turned on, patient IDs can be enabled, disabled.

Sound can be enabled, disabled.

Graph includes analyte, multiple results (values), units (mg/dL), data, meal tags, filters, goal range.

Language can be set as English (first version), Nederlands, Français, Deutsch.

Time can be changed, time format can be set as 12-hour, 24-hour.

Normal ranges can be customized.

Units per volume measured in mg/dL, mmol/L, etc.

Graphs can be measured in units per volume.

Settings include language, units per volume, graphing and tagging, patient ID.

Patient IDs can be turned off, graphing and tagging can be enabled, disabled.

Graphing and tagging can be turned on, patient IDs can be enabled, disabled.

Sound can be enabled, disabled.

Graph includes analyte, multiple results (values), units (mg/dL), data, meal tags, filters, goal range.
iNat: Nasal Flu Meter

- Google / Experience Talk / Using Conceptual Models in Interaction Design
Similar primary functions

Orca performs Tests produces Results can be created, read, tagged as deleted (automatically after memory is full) stored in

Test Type Analyte(s) Analyte Value(s) Meal Tag Date + Time Patient ID Device ID
Strip ID Upload Flag Message QC Status

iNat performs Tests produces Results can be created, read, tagged as deleted (automatically after memory is full) stored in

Printed Flu A Result (+ or -) Flu B Result (+ or -) Date + Time User ID Patient ID Device ID Test ID 1 Test ID 2 QC Status
Different analytes

**Orca**

- measure → Analytes include →
  - (First version of Orca will include)
    - Glucose
    - HDL
cLDL
- Total Cholesterol
- Triglycerides
  - (Future version of Orca may include)
  - 1,5 Ag
  - A1c
  - ALT/AST
  - BNP
  - Creatinine
  - Glycated Albumin
  - Ketones
  - White Blood Cell (WBC)
  - INR
  - hs-CRP
  - or combination of above (panel)

**iNat**

- detects → Influenza can be →
  - Type A
  - Type B
Different communication protocols

**Orca**
- has Communication Abilities
  - can be
  - FitLinx

- has Settings

- is used by
  - Patient (Primary)
  - Physician
  - Physician Assistant
  - Lab Tech

**iNat**
- has Communication Abilities
  - can be
  - Ethernet (Data Manager)
  - USB (USB Key)

- has Preferences
  - can be
  - Accessed By All Users

- has Settings
  - can be
  - Accessed By Administrators

- is used by
  - Physician
  - Physician Assistant
  - Lab Tech
Different user types

**Orca**
- **has** Communication Abilities
  - can be
  - FitLinxx
- **has** Settings
- **is used by** Patient (Primary)
  - Physician
  - Physician Assistant
  - Lab Tech

**iNat**
- **has** Communication Abilities
  - can be
  - Ethernet (Data Manager)
  - USB (USB Key)
- **has** Preferences
  - can be
  - Accessed
    - By All Users
- **has** Settings
  - can be
  - Accessed
    - By Administrators
- **is used by** Physician
  - Physician Assistant
  - Lab Tech
CMs (prior) are not IAs (below)
Other examples
Google / Experience Talk / Using Conceptual Models in Interaction Design

Android, PIM

- Contacts
  - Contents
    - Contacts
      - Creator
      - Novel
      - Read (opened)
      - Updated
      - Deleted
      - Imported
    - Sent to Cloud
    - Tagged
    - Filtered
    - Sorted
    - Shared via Email
  - Other Name or Email address or Phone number plus Source (could be Amazon cloud)
    - may contain
    - may be added manually (preferences enable auto-add to)
    - may be viewed on a
    - may be converted to

- Calendar
  - Contents
    - Events
      - Creator
      - Viewed by
      - Updated
      - Deleted
      - Sent to Cloud
      - Tagged
      - Filtered
      - Sorted
      - cannot include other objects
    - Event name
    - Event description, notes
    - Invitee names
      - Accepted, Maybe, Rejected, No response
    - Event location
      - Reminder (how far ahead, how often)
    - Source
    - may generate
      - Invitations
        - can be
        - Sort
        - Accepted
        - Rejected

- Task List
  - Contents
    - Tasks
      - Creator
      - Updated
      - Added to calendar
      - Sent to Cloud
      - Tagged (priority, type)
      - Filtered
      - Sorted
      - Marked complete
      - cannot include other objects
    - Task name
    - Source
    - may contain
    - Description
      - Deadline
      - Priority
      - Status (pending, completed)
      - Owner (other than user)
      - Dependencies

- Notes
  - Contents
    - Unstructured text
      - can be
        - Edited
        - Sent to Cloud

- Email
  - Contents
    - Email messages
      - Sent and received by Email Client
      - can be converted to

- Map
  - Contents
    - Maps
      - created
      - retrieved from or added to
      - retrieved by Map Client

Contacts List
Email Client
Tasks
Unstructured text
**Shared task manager**

<table>
<thead>
<tr>
<th>Family (only one)</th>
<th>has</th>
</tr>
</thead>
<tbody>
<tr>
<td>Members</td>
<td>can be</td>
</tr>
<tr>
<td>can be Edited</td>
<td>Viewed</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Display Names</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Edit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Password</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Created</td>
<td>Edited</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Usernames</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viewed</td>
<td>Edited</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Images</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Edit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Access Privileges (given to member)</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can add members and dependents to their</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Activity Queue</th>
<th>includes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viewed</td>
<td>Edited</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Activities</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Created</td>
<td>Edited</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Status</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Created</td>
<td>Edited</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Status</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assigned</td>
<td>- To me (default)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Status</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>In progress</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>End day (deadline)</th>
<th>has</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complete time (actual)</td>
<td>has</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Schedules</th>
<th>can have</th>
</tr>
</thead>
<tbody>
<tr>
<td>End time</td>
<td>Start day</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reminders (system level alert)</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year, Month, Day</td>
<td>Year, Month, Day, Hour, Minute, am/pm</td>
</tr>
<tr>
<td>Hour, Minute, am/pm</td>
<td>Year, Month, Day</td>
</tr>
<tr>
<td>Year, Month, Day</td>
<td>None</td>
</tr>
<tr>
<td>Year, Month, Day, Hour, Minute, am/pm</td>
<td>Bi-weekly, Monthly, Yearly</td>
</tr>
<tr>
<td>None</td>
<td>On scheduled end time</td>
</tr>
<tr>
<td>Minus 5 mins</td>
<td>Minus 15 mins</td>
</tr>
<tr>
<td>Minus 30 mins</td>
<td>Minus 1 hour</td>
</tr>
<tr>
<td>Minus 2 hours</td>
<td>Minus 2 days</td>
</tr>
<tr>
<td>Minus 2 days Snoozed (add 10 minutes)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Occurrences</th>
<th>values entered in Activity fields</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Requests</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Created</td>
<td>Edited</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Requests</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waiting</td>
<td>Answered</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Status</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted</td>
<td>- Rejected</td>
</tr>
<tr>
<td>---</td>
<td>- Rejected by requestor</td>
</tr>
<tr>
<td>---</td>
<td>- Accepted by requestor</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Status</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted</td>
<td>- Not completed</td>
</tr>
<tr>
<td>---</td>
<td>- If past deadline</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Status</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Completed</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Benefits</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Created</td>
<td>Edited</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Benefits</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amount</td>
<td>Name [text]</td>
</tr>
<tr>
<td>- Units [text]</td>
<td>Default [number]</td>
</tr>
<tr>
<td>List</td>
<td>Name [text]</td>
</tr>
<tr>
<td>Notes</td>
<td>Name [text]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Notes</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Created</td>
<td>Edited</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Notes</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amount</td>
<td>Name [text]</td>
</tr>
<tr>
<td>- Units [text]</td>
<td>Default [number]</td>
</tr>
<tr>
<td>List</td>
<td>Name [text]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Notes</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Created</td>
<td>Edited</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Notes</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amount</td>
<td>Name [text]</td>
</tr>
<tr>
<td>- Units [text]</td>
<td>Default [number]</td>
</tr>
<tr>
<td>List</td>
<td>Name [text]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scale</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Created</td>
<td>Edited</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scale</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low name [text]</td>
<td>Low value [text]</td>
</tr>
<tr>
<td>High name [text]</td>
<td>High value [text]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scale</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Created</td>
<td>Edited</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scale</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low name [text]</td>
<td>Low value [text]</td>
</tr>
<tr>
<td>High name [text]</td>
<td>High value [text]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Access Privileges (set by member)</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>View</td>
</tr>
<tr>
<td>Enter</td>
<td>Modify</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Priority?</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>Normal</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Location?</th>
<th>can be</th>
</tr>
</thead>
<tbody>
<tr>
<td>City</td>
<td>State</td>
</tr>
</tbody>
</table>

| Occurrences | values entered in Activity fields |

---

Google / Experience Talk / Using Conceptual Models in Interaction Design
Flipboard, social magazine

Flipboard provides a Pool that changes continually. Articles can be displayed, read, saved, shared, grouped, or flagged (inappropriate). Articles can appear in the Pool and disappear from the Pool.

Stories may include text, images, videos, comments, and other elements.

Flipboard Articles can be displayed, read, rated, saved, shared, grouped, flagged (as inappropriate), and may appear in the Pool or disappear from the Pool.

Flipboard provides a pool of stories, posts, images, videos, comments, and allows users to mark articles as favorites on Twitter.

Users can customize their experience by changing text and font size, as well as adjusting the display of images, videos, and comments.

Users can update their account information, including name, email, password, and photo.

Flipboard includes about, settings, and help sections.

User sources include Facebook, Twitter, Google Reader, LinkedIn, Instagram, Flickr, Tumblr, and Sina Weibo.

* Cover Stories appear differently than other content—they appear on the "cover" of Flipboard, and cycle through full-screen images with article titles. Titles are not clickable.
Zite, discovery engine
PLX, early data model study

Dashboard
Annotations
Filter Widgets
Metadata

Container
Annotations
Filter Widgets
Metadata

View
Query
Proximate
Expression
SELECT columns A, C, D
Source
FROM table foo-2
Final
Expression
SELECT column (processes that led to A, C, D, E)
Source
FROM table foo-root
Schedule
BEGIN every Sunday at 23:00:00 GMT
UNTIL ongoing
Transforms
Joins
Derivations
SET column F = C + D + E
Filters
WHERE country = CN or JP or KR

Table foo-2

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cnty</td>
<td>Region</td>
<td>Prod C</td>
<td>Prod D</td>
<td>Prod E</td>
</tr>
<tr>
<td>1</td>
<td>CN</td>
<td>25</td>
<td>38</td>
<td>19</td>
</tr>
<tr>
<td>2</td>
<td>JP</td>
<td>37</td>
<td>42</td>
<td>33</td>
</tr>
<tr>
<td>3</td>
<td>KR</td>
<td>27</td>
<td>12</td>
<td>41</td>
</tr>
<tr>
<td>4</td>
<td>TW</td>
<td>13</td>
<td>18</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>SP</td>
<td>9</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>UK</td>
<td>43</td>
<td>29</td>
<td>16</td>
</tr>
<tr>
<td>7</td>
<td>FR</td>
<td>32</td>
<td>28</td>
<td>15</td>
</tr>
</tbody>
</table>

Result
All data
Sample data
Column title list

<table>
<thead>
<tr>
<th>A</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cnty</td>
<td>Prod C</td>
<td>Prod D</td>
<td>Prod E</td>
<td>Total</td>
</tr>
<tr>
<td>1</td>
<td>CN</td>
<td>25</td>
<td>38</td>
<td>19</td>
</tr>
<tr>
<td>2</td>
<td>JP</td>
<td>37</td>
<td>42</td>
<td>33</td>
</tr>
<tr>
<td>3</td>
<td>KR</td>
<td>27</td>
<td>12</td>
<td>41</td>
</tr>
</tbody>
</table>

Visualization
Type
Bar Chart
Properties
Standard
Annotations
Text
Filter Widgets
DROPDOWN = CN, JP, KR

Metadata
Title
Quarterly Revenue
Keywords
Filtered by China, Japan, Korea
Description
Natali
Creator
August 10, 2011
Date Created
Jeff
Permissions
Owners
Collaborators
Readers
Mike
David

Other Views

Other Containers
Other Views
Summary
Product teams should agree on a conceptual model—a definition of what users need to know—well before creating wire-frames and writing code.

The model will change over the course of the product development process.

Developing a standard form for conceptual models will make them easier to understand, easier to make, and easier to teach new designers.
Special thanks to
Jonathan Arnowitz
Bryan Crowe
Austin Henderson
Marci Robidoux
Mathew Varghese

hugh@duberly.com
415 648 9799

Presentation posted at
www.duberly.com/presentations/conceptual_models.pdf