Model Aspects

Dr. Ernest Cachia
Model Views

(1/3)

Example:
A simulation system
Model Views (2/3)

Example:
A process control system
Model Views (3/3)

Example:
An information system
Example of System Model Aspects

• **System:** Telephone set

• **Information Aspects**
  – **Functional:**
    1) Allow call selection
    2) Provide voice data
    3) Sound audio signal; *etc.*
  – **Dynamic:**
    1) If a call is incoming and device is idle, sound signal
    2) If the handset is raised the device is set to busy; *etc.*
  – **Data:**
    1) Subscriber number format (local, foreign, DDI, *and so on*)
    2) Signal standards; *etc.*
Models and their Main Tools

<table>
<thead>
<tr>
<th>Model / Aspect</th>
<th>Supporting tool</th>
<th>Clarification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functional</td>
<td>DFD / PSpec / DD</td>
<td>DSD, BNF or text as PSpec held as a DD</td>
</tr>
<tr>
<td>Data</td>
<td>ERD / DD</td>
<td></td>
</tr>
<tr>
<td>Dynamic</td>
<td>STD / ELH</td>
<td>STD can also be called FSM in some literature</td>
</tr>
</tbody>
</table>
Model Relationships

- **Data entity relations**
  - DFD
  - PSpec
  - STD/FSM

- **Data transformations (flows & stores)**
  - ERD
  - DD
  - ELH

- **Static data on flows & in stores**
  - DSD
    - BNF
    - Textual
Taking stock

• DFDs – Done!
• STDs/FSMs – Done!
• DSDs – Done!
• PSpecs – Done!
• DDs – A concept not a notation as such.
• BNF – Will do something similar shortly in DD context.
• ELHs – Still to come.
Data Dictionary Concepts

- Centralised repository
- Synchronisation point
- Focal point
- Data sharing
- Standardised (or standardise-able) data exchange
- Backup
- Verifiability
Notation used to show DD structure (very simplified BNF)

= is composed of
+
++ and
[||] either...or
{ }\(^n\) \(n\) repetitions of
( ) optional data
* * comment delimiters
Example format of data item
(Idea adapted from “Software Engineering Fundamentals” by A. Behforooz et al.)

Data item name is “UoM-telephone-number”

Structure of UoM-telephone-number is as follows:

Telephone_number= [local_extension|outside_number]
Local_extension = [200|201|...400|2000|2001|...3000]
Outside_number = line_access + local_number
Line_access = 9
Local_number = [2|3|4|5|6|8] + rest_of_number
Rest_of_number = *Any five-digit sequence*
Example format of data item "dfd_entity"

Structure of data item "dfd_entity" could be as follows:

- Node_ID
- Node_type
- Node_label
- Node_fullname
- System_name
- Diagram_name
- Diagram_level
- Input_links
- Output_links
- Parent_node_ID
- Parent_diagram
- Flags (e.g. Decomposed, PSPEC, undefined, has annotation, auto- or manually created, etc.)
- Directory_location
- Date of creation
- Date of last edit
- Date of last view
- Creator/maintainer
- Contact_details
PSpec Example

Get Cust Name, Cust Addr and Cust Tel No.;

Get Account No.;

Customer Record := [Cust Name, Cust Addr, Cust Tel No.];

Write Customer Record to Customers;

Balance := 0;

Account Record := [Account No., Balance, Curr Date];

Write Account Record to Accounts;
Another DD Example

Registration_Details = Cust Name + Cust Addr + Cust Tel No. + Account No.

Cust_Details = Cust Name + Cust Addr + Cust Tel No.

Accounts = {Account}

Account = Account No. + Balance + Date Opened
Entity Life Histories (ELHs)

- Meant to model the “stages” a data entity can go through (i.e. how it can be used)
- Is based on what is known as an Event-Entity Matrix (EEM)
- Quite intuitive to conceptualise and subsequently construct
- Form the basis of other, more specific, system models
“First-shot” ELH Example
Steps to Construct an ELH

- Identify events
- Produce event/entity matrix
- Produce initial ELHs
- Check initial ELHs
- Review ELHs
- Add operations to ELHs
- Add state indicators to ELHs
Start off with an EEM

<table>
<thead>
<tr>
<th>Event</th>
<th>Title</th>
<th>Reader</th>
<th>Book Copy</th>
<th>Loan</th>
<th>Reservation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lend</td>
<td>M</td>
<td>M</td>
<td>C</td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>Send Reminder1</td>
<td>M</td>
<td></td>
<td></td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Send Reminder2</td>
<td>M</td>
<td></td>
<td>M</td>
<td></td>
<td>C</td>
</tr>
<tr>
<td>Reserve</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Return</td>
<td>M</td>
<td>D*</td>
<td>M</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Register reader</td>
<td>C</td>
<td></td>
<td></td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>Deregister reader</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add book</td>
<td>C (First)</td>
<td></td>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Remove book</td>
<td>D (Last)</td>
<td></td>
<td>D</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Produce an ELH

- Reader
  - Register
  - Mid-life
    - *
      - Lend
      - Send Reminder 1
      - Send Reminder 2
  - Deregister
Check the ELH

• For completeness
  – Did we identify and cater for all the events? E.g.
    • Which events create new attributes?
    • Which events change attributes?
    • Which events setup or remove relationships with “master” entities?

• For consistency
  – Will data relationships still be valid in the light of change? E.g.
    • What will happen to a “detail” if its “master” dies?
    • What will happen to a “master” if its “details” are all removed?
Death of a “Master”
Death of the Last “Detail”