Learning from Past Trial and Error:

Some History of Reverse Engineering to Requirements

Elliot Chikofsky
RETR Workshop at WCRE 2005
Pittsburgh PA

e.chikofsky@computer.org
+1 781-272-0049

Some History

– “Requirements Engineering” tools; “Systems Analysis and Design”
  – PSL / PSA (Univ Michigan ISDOS)
  – RSL - REVS - SREM (TRW)

– Computer-Aided Software Engineering (CASE)
  – Excelerator; KnowledgeWare; Bachman; Popkin

– IBM’s AD/Cycle initiative
  – IBM Repository; Enterprise Model
  – Fight over the inclusion of a Requirement object
**WCRE-inventing the Wheel**

- Circa 1980 [PSL/PSA; mainframe CASE]:
  - Rev Engr from Object Code
  - Automatic documentation by code reverse engr
  - Data collection from many kinds of multiple artifacts
  - Process matching defect reports and software code artifacts found to recovered architecture framework

**Levels of Abstraction**

*Figure 1.* Relationship between terms. Reverse engineering and related processes are transformations between or within abstraction levels, represented here in terms of life-cycle phases.

Chikofsky and Cross (1990)
What we Can and Cannot see

- **Design Issues**
  - Alternatives rejected
  - Ramifications of decisions

- **Existing design Code**

- **Unplanned ramifications (side effects)**

Forward Engineering

Reverse Engineering

Unplanned ramifications (side effects)

Chikofsky and Cross (1990)

---

CASE Intent: Evolutionary Development

Circa 1986 [Excelerator / CASE]

- **Design**
- **Generate System**
- **Reverse Engineer**
- **Analysis**

**REPOSITORY**

**Reverse Engineering to Requirements**

A Permanent, Automatic part of the Process
Evolution of WHY to Reverse Engineer

- Software Maintenance Support
- CASE for Existing Systems
- Design Recovery
- Migration to <fill-in-your-favorite>
- Program Understanding / Comprehension
- Year 2000
- Asset Recovery
- Defensive / Offensive Obfuscation

Engineering Processes

ref. Index Technology, early 1990s
Do we understand original constraints?

“If we do not bother taking soil samples, we can save 5,000 Lira and two weeks.”

RETR Challenges

- What kind of requirements?
  - Functional Requirements; Engineering Requirements;
  Interface Requirements; Environmental Requirements
- Getting from Physical models to Logical models
- Whose requirements are they?
  - Achieving meaning for the functional users
- Coping with incompleteness
- Understanding the ramifications of not knowing the original constraints
American Flange

ORDER ENTRY & INVOICING SYSTEM

ORDERS

SALES FORCE INFO

INVOICES

PRODUCT CATALOG

Data Structures

<table>
<thead>
<tr>
<th>Data Structure</th>
<th>Field(s)</th>
<th>Type</th>
<th>Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SALESFORCE-FILE</td>
<td>SALESPERSON-REC</td>
<td>Primary Key</td>
<td>String 30</td>
<td>Salesperson-Name</td>
</tr>
<tr>
<td></td>
<td>COMMISSION-RATE</td>
<td>---</td>
<td>Real</td>
<td>Commission-Rate</td>
</tr>
<tr>
<td></td>
<td>SALESPERSON-NAME</td>
<td>---</td>
<td>String 30</td>
<td>Salesperson-Name</td>
</tr>
<tr>
<td>INVOICES-FILE</td>
<td>INVOICE-REC</td>
<td>Primary Key</td>
<td>Integer</td>
<td>Invoice-Number</td>
</tr>
<tr>
<td></td>
<td>AMOUNT</td>
<td>---</td>
<td>Real</td>
<td>Amount</td>
</tr>
<tr>
<td></td>
<td>DATE-PAID</td>
<td>---</td>
<td>Date</td>
<td>Date-Paid</td>
</tr>
<tr>
<td></td>
<td>ORDER-NUMBER</td>
<td>Foreign Key</td>
<td>Integer</td>
<td>Order-Number</td>
</tr>
<tr>
<td></td>
<td>SALESPERSON-NAME</td>
<td>Foreign Key</td>
<td>String 30</td>
<td>Salesperson-Name</td>
</tr>
<tr>
<td>ORDERS-FILE</td>
<td>ORDER-REC</td>
<td>Primary Key</td>
<td>Integer</td>
<td>Order-Number</td>
</tr>
<tr>
<td></td>
<td>CUSTOMER-NAME</td>
<td>---</td>
<td>String 30</td>
<td>Customer-Name</td>
</tr>
<tr>
<td></td>
<td>ADDRESS</td>
<td>---</td>
<td>String 90</td>
<td>Address</td>
</tr>
<tr>
<td></td>
<td>ORDER-DATE</td>
<td>---</td>
<td>Date</td>
<td>Order-Date</td>
</tr>
<tr>
<td></td>
<td>CUSTOMER-NUMBER</td>
<td>---</td>
<td>Combined</td>
<td>Customer-Number</td>
</tr>
<tr>
<td></td>
<td>CUST-LETTER</td>
<td>---</td>
<td>String 1</td>
<td>CUST-LETTER</td>
</tr>
<tr>
<td></td>
<td>CUST-SEQ</td>
<td>---</td>
<td>Integer 4</td>
<td>CUST-SEQ</td>
</tr>
<tr>
<td>LINE-ITEM</td>
<td>ITEM-REC</td>
<td>Part of Key; Foreign Key</td>
<td>Integer</td>
<td>Item-Number</td>
</tr>
<tr>
<td></td>
<td>ORDER-NUMBER</td>
<td>Part of Key; Foreign Key</td>
<td>Integer</td>
<td>Order-Number</td>
</tr>
<tr>
<td></td>
<td>QUANTITY</td>
<td>---</td>
<td>Integer</td>
<td>Quantity</td>
</tr>
<tr>
<td></td>
<td>PRICE</td>
<td>---</td>
<td>Real</td>
<td>Price</td>
</tr>
<tr>
<td>CATALOG-FILE</td>
<td>CATALOG-REC</td>
<td>Primary Key</td>
<td>Integer</td>
<td>Item-Number</td>
</tr>
<tr>
<td></td>
<td>SUPPLIER</td>
<td>---</td>
<td>String 30</td>
<td>Supplier</td>
</tr>
<tr>
<td></td>
<td>DESCRIPTION</td>
<td>---</td>
<td>String 50</td>
<td>Description</td>
</tr>
<tr>
<td></td>
<td>COST</td>
<td>---</td>
<td>Real</td>
<td>Cost</td>
</tr>
</tbody>
</table>
Data Model

SALESPERSON
- Name
- Commissions
- Rate

INVOICE
- Invoice #
- Amount
- Date Paid

ORDER
- Order #
- Date
- Customer #
- Customer Name
- Address

LINE ITEM
- Item # + Order #
- Quantity
- Price

CATALOG
- Item #
- Supplier
- Description
- Quantity on Hand
- Cost

Pricing Model

ORDER

LINE ITEM
- Price

CATALOG
Sales & Order Management

Catalog

Copyright © 2003 E.Chikofsky
Engineering Processes

Business or Engineering Need → Requirements Engineering → Architecture and Design → Implementation (Code Generation) → Operational System → Reverse Engineering → Existing Systems

ref. Index Technology, early 1990s