System Engineering

Software Engineering 10
**Business Process Engineering**

- Uses an integrated set of procedures, methods, and tools to identify how information systems can best meet the strategic goals of an enterprise
- Focuses first on the enterprise and then on the business area
- Creates enterprise models, data models and process models
- Creates a framework for better information management distribution, and control

---

**BPE Hierarchy**

```
The enterprise

Information Strategy Planning (World view)
```

```
Business Area Analysis (Domain view)
```

```
Business System Design (Element view)
```

```
Construction & Integration (Detailed view)
```

```
A business area
```

```
A business area
```

```
A business area
```

```
A business area
```

```
Software engineer
```

```
Processing requirement
```

---
Information Strategy Planning

Management issues
• define strategic business goals/objectives
• isolate critical success factors
• conduct analysis of technology impact
• perform analysis of strategic systems

Technical issues
• create a top-level data model
• cluster by business/organizational area
• refine model and clustering

Defining Objectives and Goals

Objective-general statement of direction
Goal-defines measurable course of action

Examples:
• objective-reduce manufactured cost of our product
• goals
  * decrease reject rate by 20% in first 6 months
  * gain 10% price concessions from suppliers
  * re-engineer 30% of components for ease of manufacture during first year

Objectives tend to be strategic while goals tend to be tactical
**Business Area Analysis**

Define "naturally cohesive groupings of business functions and data" (Martin)

Perform many of the same activities as ISP, but narrow scope to individual business area

Identify existing (old) information systems and determine compatibility with new ISP model

- define systems that are problematic
- defining systems that are incompatible with
- new information model
- begin to establish re-engineering priorities

---

**The BAA Process**

```
<table>
<thead>
<tr>
<th>admin.</th>
<th>manufacturing</th>
</tr>
</thead>
<tbody>
<tr>
<td>sales</td>
<td>QC</td>
</tr>
<tr>
<td>acct</td>
<td>distribution</td>
</tr>
<tr>
<td></td>
<td>engineering</td>
</tr>
</tbody>
</table>
```

- Process Flow Models
- Data Model
- Process Decomp. Diagram
- Matrices e.g. entity/process matrix
CASE and Business Process Engineering: Data oriented Tools

• Supporting information engineering by James Martin and others.
  • Data is seen as the main resource of the enterprise.
  • Based on data-models defining the basic data units and data relationship for the entire enterprise. Databases and process logic are derived from these models.

• Information Engineering Facility (IEF), Information Engineering Workbench (IEW), ER-Designer (ERD).

Information Engineering Facility (IEF)

• Enforces top-down planning, analysis, design, and implementation.
• Primarily used for developing on-line/batch, screen-oriented administrative systems.
• Supports developing windows-based applications.
• Strong separation of database and process-logic.
• Information planning, analysis and design is done on a workstation, after relevant parts of the entire data model has been downloaded from main-frame.
• Compilation of modules, database generation and code generation takes place on the mainframe.
IEF - Information planning

Entity Types

<table>
<thead>
<tr>
<th>Entity Type</th>
<th>Customer</th>
<th>Order</th>
<th>OrderLine</th>
<th>Product</th>
<th>Part</th>
<th>Supplier</th>
<th>Warehouse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Functions</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Customer management</td>
<td>R</td>
<td>U</td>
<td>D</td>
<td>R</td>
<td>U</td>
<td>D</td>
<td>R</td>
</tr>
<tr>
<td>Change Order</td>
<td>C</td>
<td>U</td>
<td>D</td>
<td>C</td>
<td>U</td>
<td>D</td>
<td>C</td>
</tr>
<tr>
<td>Cancel Order</td>
<td>R</td>
<td>U</td>
<td>D</td>
<td>R</td>
<td>U</td>
<td>D</td>
<td>R</td>
</tr>
</tbody>
</table>

Matrices

IEF - Analysis

Entity relationship diagrams
IEF - Analysis II

Running the company
  Marketing
  Selling
    Customer registration
    Order processing
      Accept order
      Change order
      Cancel order
  Producing

Process hierarchy

IEF - Analysis III

Order request
  Order inf
    Accept order
      Order inf
        Available products

Process dependencies
**IEF - Analysis: Process-handling**

**Process:**
- ACCEPT ORDER

**Imports:**
- Entity View to_be_ordered_product

**Exports:**
- Entity View confirmed_order_line

**Entity Actions:**
- Entity View confirmed_order_line

- READ to_be_controlled_product
  - WITH name EQUAL TO to_be_ordered_product.name
  - WHEN not found

- CREATE confirmed_order
  - SET date TO "system date"
  - SET number TO "next free value"
  - ASSOCIATE WITH to_be_controlled_customer WHICH places IT
  - ASSOCIATE WITH confirmed_order_line WHICH details IT
  - WHEN already exists

- MOVE confirmed_order TO accepted_order,

---

**IEF - Design: Dialog-flow**

**Menu**
- Accept order
- Accept order header
- Accept order lines
- Accept customer

Business Process Engineering
**IEF - Design: Screen design**

**TRANCODE**

**ORDER PROCESSING**

ORDER NUMBER: 9999999
CUSTOMER NUMBER: 9999999
NAME: XXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXX XX XXXXXXXXX
XXXXXXXXXXXXXXXX

ORDER DATE: MM-DD-YY
STATE: XX

<table>
<thead>
<tr>
<th>LINE</th>
<th>PRODUCT</th>
<th>DESCRIPTION</th>
<th>QTY</th>
<th>UN</th>
<th>UNIT PRICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>999</td>
<td>9999999</td>
<td>XXXXXXXXXXXXXXXXXXXXXXXXXXX</td>
<td>9999</td>
<td>XX</td>
<td>$88.89</td>
</tr>
<tr>
<td>999</td>
<td>9999999</td>
<td>XXXXXXXXXXXXXXXXXXXXXXXXXXX</td>
<td>9999</td>
<td>XX</td>
<td>$88.89</td>
</tr>
<tr>
<td>999</td>
<td>9999999</td>
<td>XXXXXXXXXXXXXXXXXXXXXXXXXXX</td>
<td>9999</td>
<td>XX</td>
<td>$88.89</td>
</tr>
<tr>
<td>999</td>
<td>9999999</td>
<td>XXXXXXXXXXXXXXXXXXXXXXXXXXX</td>
<td>9999</td>
<td>XX</td>
<td>$88.89</td>
</tr>
</tbody>
</table>

<<<ERR>>> <<<ERR>>> <<<ERR>>> <<<ERR>>> <<<ERR>>> <<<ERR>>> <<<ERR>>> <<<ERR>>> <<<PFK>>> <<<PFK>>> <<<PFK>>> <<<PFK>>> <<<PFK>>> <<<PFK>>> <<<PFK>>> <<<PFK>>>

**IEF: From Analysis to Code**

Information Strategy Planning
Business Area Analysis
Business System Design
Technical Design
Database generation
Code generation

[Diagram showing the process from Analysis to Code with specific areas marked as 1-5, 6-11, and 12-16]
**IEF: From Analysis to Code**

**Information Strategy Planning:**
1. Matrix Processor
2. Organizational Hierarchy Diagram
3. Subject Area Diagram
4. Function Hierarchy Diagram
5. Function Dependency Diagram

**Business System Design:**
12. Dialog Flow Diagram
13. Screen Design
14. Prototyping
15. Procedure Action Diagram
16. Structure Chart

**Business Area Analysis:**
6. Entity Relationship Diagram
7. Process Hierarchy Diagram
8. Process Dependency Diagram
9. Process Action Diagram
10. Structure Chart
11. Matrix Processor

**Technical Design:**
17. Data Structure Diagram

---

**Product Engineering**

- **System analysis** (World view)
- **Component engineering** (Domain view)
- **Analysis & Design Modeling** (Element view)
- **Construction & Integration** (Detailed view)

**The complete product**

- **hardware**
- **software**

**Processing requirement**

- **data**
- **function**
- **behavior**

**Program component**

**Software engineer**

---

Business Process Engineering
Requirements Engineering

• Elicitation — determining what the customer requires
• Analysis & negotiation — understanding the relationships among various customer requirements and shaping those relationships to achieve a successful result
• Requirements specification — building a tangible model of requirements

Requirements Engineering

• System Modeling — building a representation of requirements that can be assessed for correctness, completeness, and consistency
• Validation — reviewing the model
• Management — identify, control and track requirements and the changes that will be made to them
**System Allocation**

- objects
- processes
- performance
- constraints

Allocation

- software
- hardware
- people
- data
- documents
- procedures

system components

support infrastructure

**System Model Template**

<table>
<thead>
<tr>
<th>user interface processing</th>
<th>input processing</th>
<th>process and control functions</th>
<th>output processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>maintenance and self-test</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Product Engineering
A Conveyor Line Sorting System (CLSS).

Conveyor line motion

Bar code

- Read bar code input
- Read pulse tachometer
- Decode part code data
- Do database look-up
- Determine bin location
- Produce control signal for shunt
- Maintain record of box destinations

System Context Diagram for CLSS
**System Flow Diagram**

- **Operator Requests**
  - **Bar Code Reader Subsystem**
    - **Bar Code**
    - **Raw Bar Code Data**
  - **Sensor Data Acquisition Subsystem**
    - **Pulse Tach Input**
  - **Data Base Access Subsystem**
    - **Data Base**
  - **Diagostics Subsystem**
  - **CLSS Processing & Control**
    - **Part Number**
    - **Line Speed**
    - **Bin Location**
    - **Key**
    - **Sort Records**
    - **Timing/Location Data**
    - **Shunt Control Status**
  - **CLSS Queries, Reports, Displays**
    - **Operator Interface Subsystem**
    - **Shunt Control Subsystem**
    - **Report Formatting Subsystem**
    - **Mainframe Communic. Driver**
    - **Report Formatting Data**

**The Transition to Software Engineering**

- **Allocation**
  - **Software**
  - **Hardware**
  - **People**
  - **Data**
  - **Documents Procedures**

Software Engineering begins ...

- Structured Analysis
- Structured Design
**Gathering Requirements**

**Responsibilities**
- Key Point: The Customer is responsible for the requirements.
- Programmers help to gather and clarify requirements. Customers especially need help with non-functional requirements and with working out the details of acceptance tests.

**Documentation**
- User Stories
- Acceptance Test Cases
User Stories

A short description of the behavior of the system from the point of view of the Customer
Use the Customer’s terminology without technical jargon
One for each major feature in the system
Must be written by the users
Are used to create time estimates for release planning
Replace a large Requirements Document
**User Stories continued**

Drive the creation of the acceptance tests:

- Must be one or more tests to verify that a story has been properly implemented

Different than Requirements:

- Should only provide enough detail to make a reasonably low risk estimate of how long the story will take to implement.

Different than Use Cases:

- Written by the Customer, not the Programmers, using the Customer’s terminology
- More “friendly” than formal Use Cases

---

**User Stories continued**

User stories have three crucial aspects:

**Card**

- Enough information to identify the story

**Conversation**

- Customer and Programmers discuss the story to elaborate on the details
- Verbal when possible, but documented when required

**Confirmation**

- Acceptance tests to confirm that the story has been properly implemented
**User Story Examples**

A user wants access to the system, so they find a system administrator, who enters in the user's First Name, Last Name, Middle Initial, E-Mail Address, Username (unique), and Phone Number.

*Risk: Low  
Cost: 2 points*

**User Story Examples continued**

The user must be able to search for a book.

*Risk: High  
Cost: (too large!)*
User Story Examples continued

The user must be able to search for a book by Title, and display the results as a list.

Risk: Med.       Cost: 1 point

User Story Examples continued

The user must be able to search for a book by Author, and display the results as a list.

Risk: Med.       Cost: 1 point
User Story Examples continued

The user must be able to search for a book by ISBN number, and display the results as a list.

Risk: Med.       Cost: 1 point

User Story Examples continued

The user must be able to search for a book by Category, and display the results as a list.

Risk: Med.       Cost: 2 points
Acceptance Tests

- Formal test to determine if a system satisfies its acceptance criteria, i.e. the User Stories
- At least one Acceptance Test for each Story
- User story is not complete before succeeding its acceptance tests.

Acceptance Tests 2

- New tests for each iteration, or the development will report zero progress.
- A story may have one or many acceptance tests.
- Should be automated to be run often.
- The XP team schedules time to fix any failed tests for each iteration.
**Release Planning**

- Estimate a Story (Programmer)
- Split a Story (Customer)
- Spike a Story (Programmer)
- Declare Velocity (Programmer)
- Choose Scope (Customer)
- Sort Stories by Value (Customer)

**Exploration**  **Planning**

---

**Morgan - metaphors for thinking about organisations**

1. goal-seeking **machine** with interchangeable parts
2. biological **organism** that continually adapts to change
3. central **brain** that can respond to, and predict, change
4. centring on a set of shared **values** and beliefs,
5. centring on **power** and conflict, as a means whereby individuals achieve their own aspirations or mutual self-interest,
6. centring on norms of behaviour, so that the organisation is likened to a psychic **prison**
7. flux and **transformation**
8. instrument of **domination**

Gareth Morgan, Images of Organisation