C H A P T E R

9

PROCESS MODELING

Chapter Nine   Process Modeling

- Define systems modeling and differentiate between logical and physical system models.

- Define process modeling and explain its benefits.

- Recognize and understand the basic concepts and constructs of a process model.

- Read and interpret a data flow diagram.

- Explain when to construct process models and where to store them.

- Construct a context diagram to illustrate a system's interfaces with its work environment.

- Identify use cases, external and temporal business events for a system.

- Perform event partitioning and organize events in a functional decomposition diagram.

- Draw event diagrams and merge those events into a system diagram.

- Draw primitive data flow diagrams and describe the elementary data flows and processes in terms of data structures and procedural logic (Structured English and decision tables), respectively.

- Document the distribution of processes to locations.

- Synchronize data and process models using a CRUD matrix.

Models: Logical and Physical

Model – a pictorial representation of reality.

Just as a picture is worth a thousand words, most models are pictorial representations of reality.

Logical model – a nontechnical pictorial representation that depicts what a system is or does. Synonyms are *essential model*, *conceptual model*, and *business model*.

Physical model – a technical pictorial representation that depicts what a system is or does and how the system is implemented. Synonyms are *implementation model* and *technical model*.

Why Logical System Models

- Logical models remove biases that are the result of the way the system is currently implemented, or the way that any one person thinks the system might be implemented.

- Logical models reduce the risk of missing business requirements because we are too preoccupied with technical results.

- Logical models allow us to communicate with end-users in nontechnical or less technical languages.
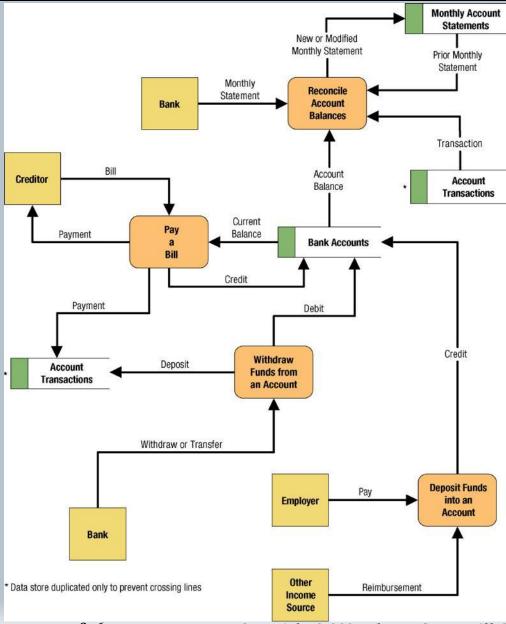
Process Modeling and DFDs

Process modeling – a technique used to organize and document a system's processes.

- – Flow of data through processes
- – Logic
- – Policies
- – Procedures

Data flow diagram (DFD) – a process model used to depict the flow of data through a system and the work or processing performed by the system. Synonyms are bubble chart, transformation graph, and process model.

DFDs have become a popular tool for business process redesign.

Simple Data Flow Diagram

Differences Between DFDs and Flowcharts

- Processes on DFDs can operate in parallel (at-the-same-time)
  – Processes on flowcharts execute one at a time

- DFDs show the flow of data through a system
  – Flowcharts show the flow of control (sequence and transfer of control)

- Processes on a DFD can have dramatically different timing (daily, weekly, on demand)
  – Processes on flowcharts are part of a single program with consistent timing

Systems Thinking

Systems thinking is the application of formal systems theory and concepts to systems problem solving.
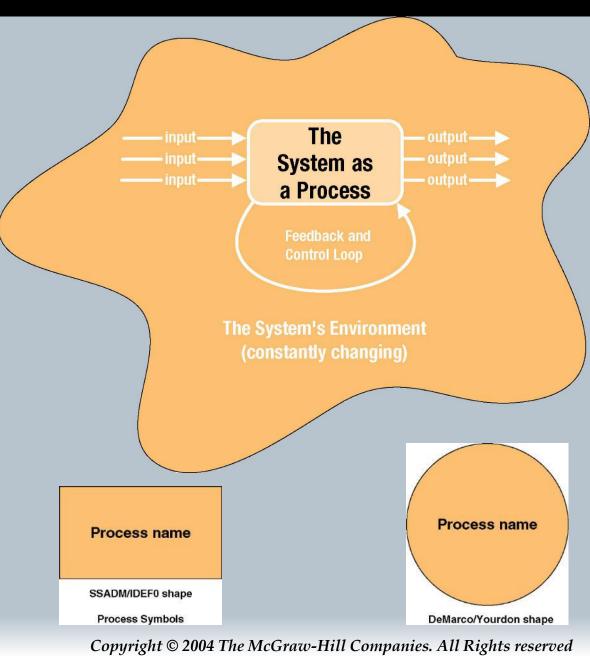
DFDs are a tool that supports systems thinking.

Process Concepts

Process – work performed by a system in response to incoming data flows or conditions. A synonym is *transform*.

input → input → input → **The System as a Process** → output → output → output

Feedback and Control Loop

The System's Environment (constantly changing)
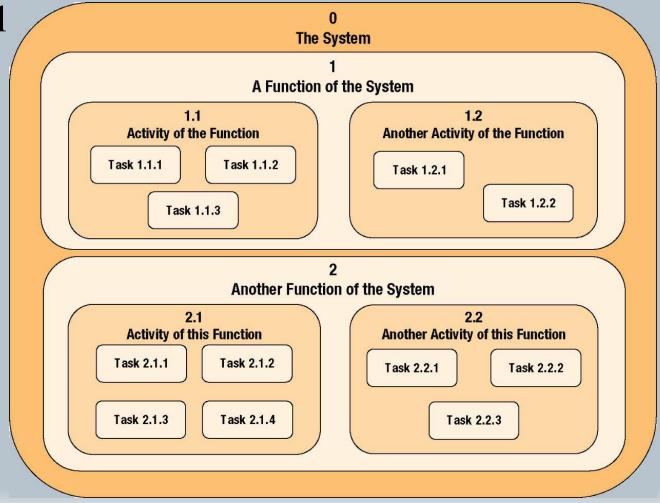
Process name

Gane & Sarson shape; used throughout this book

Process name

SSADM/IDEF0 shape

Process Symbols

Process name

DeMarco/Yourdon shape

Process Decomposition
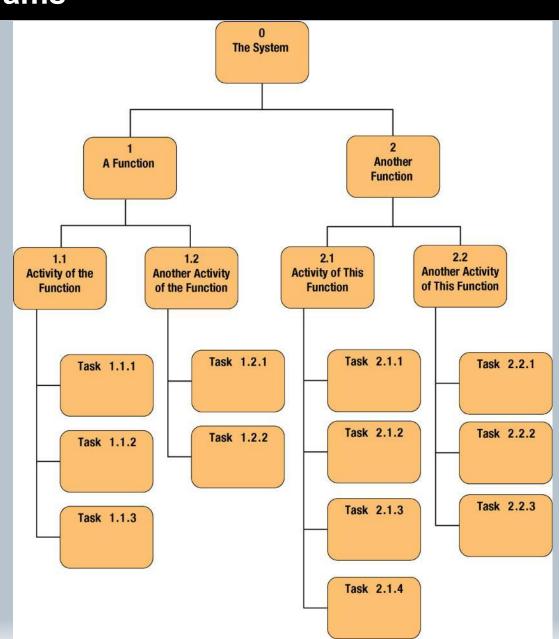
Decomposition – the act of breaking a system into sub-components. Each level of abstraction reveals more or less detail

0
The System

1
A Function of the System

1.1
Activity of the Function

Task 1.1.1   Task 1.1.2

Task 1.1.3

1.2
Another Activity of the Function

Task 1.2.1

Task 1.2.2

2
Another Function of the System

2.1
Activity of this Function

Task 2.1.1   Task 2.1.2

Task 2.1.3   Task 2.1.4

2.2
Another Activity of this Function

Task 2.2.1   Task 2.2.2

Task 2.2.3

Decomposition Diagrams

Decomposition diagram – a tool used to depict the decomposition of a system. Also called hierarchy chart.

Types of Logical Processes

Function – a set of related and ongoing activities of a business.

– A function has no start or end.
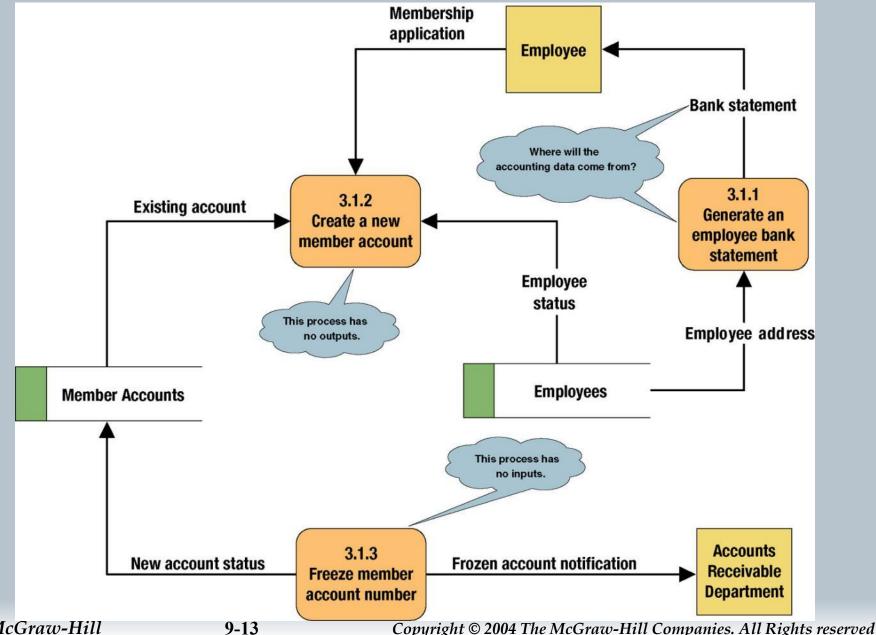
Event – a logical unit of work that must be completed as a whole. Sometimes called a *transaction*.

– An event is triggered by a discrete input and is completed when the process has responded with appropriate outputs.
– Functions consist of processes that respond to events.

Elementary process – a discrete, detailed activity or task required to complete the response to an event. Also called a *primitive process*.

– The lowest level of detail depicted in a process model.
– Should be names with a strong action verb followed by an object clause that describes what the work is perform on (or for).

Common Process Errors on DFDs

Process Logic

- Decomposition diagrams and data flow diagrams are effective tools for identifying processes, but are not good at showing the logic inside those processes.
  - Eventually need to specify detailed instructions.
  - Should effectively communicate with both users and programmers.
  - Flowcharts and pseudocode are difficult for users to understand.
  - Natural English is no imprecise (see Figure 9-6).
  - Structured English has advantages of natural English with some of the rigor of programming logic.

Problems with Natural English

- **Many of us do not write well, and we also tend not to question our writing abilities.**
- **Many of us are too educated to communicate with an audience that may not have had the same educational opportunities.**
- **Some of us write everything like it was a program. If business procedures required such precision, we'd write everything in a programming language.**
- **Too often, we allow the jargon and acronyms of computing to dominate our language.**
- **English statements frequently have an excessive or confusing scope.**
- **We overuse compound sentences**
- **Too many words have multiple definitions.**
- **Too many statements use imprecise adjectives.**
- **Conditional instructions can be imprecise.**
- **Compound conditions tend to show up in natural English.**

Source: Adapted from Matthies, Leslie, *The New Playscript Procedure,*
(Stamford, CT: Office Publications, Inc. 1977)

Structured English

Structured English – a language syntax for specifying the logic of a process.

- Based on the relative strengths of structured programming and natural English.

1. For each CUSTOMER NUMBER in the data store
   a. For each LOAN in the data store LOANS that matches the CUSTOMER NUMBER:
      1) Keep a running total of NUMBER OF LOANS for CUSTOMER NUMBER.
      2) Keep a running total of the ORIGINAL LOAN PRINCIPAL for the CUSTOMER NUMBER.
      3) Keep a running total of CURRENT LOAN BALANCE for CUSTOMER NUMBER.
      4) Keep a running total of AMOUNTS PAST DUE for the CUSTOMER NUMBER.
   b. If the TOTAL AMOUNTS PAST DUE for the CUSTOMER NUMBER is greater than $100.00
      1) Write the CUSTOMER NUMBER and all their data as described in the data flow LOANS AT RISK.

      1) Exclude the CUSTOMER NUMBER and data from the flow LOANS AT RISK.

Structured English Constructs (Part 1)

Structured English Procedural Structures

Construct	Sample Template
Sequence of steps – Unconditionally perform a sequence of steps.	[Step 1] [Step 2] … [Step n]

Simple condition steps – If the specified condition is true, then perform the first set of steps. Otherwise, perform the second set of steps.
 	
Use this construct if the condition has only two possible values.
 	
(Note: The second set of conditions is optional.)
	If [truth condition]
	    then
	        [sequence of steps or other conditional steps]
	else
	        [sequence of steps or other conditional steps]
	~~End If~~

Complex condition steps – Test the value of the condition and perform the appropriate set of steps.
 	
Use this construct if the condition has more than two values.
	Do the following based on [condition]:
	    Case 1: If [condition] = [value] then
	        [sequence of steps or other conditional steps]
	    Case 2: If [condition] = [value] then
	        [sequence of steps or other conditional steps]
	    …
	    Case n: If [condition] = [value] then
	        [sequence of steps or other conditional steps]
	~~End Case~~

Structured English Constructs (Part 2)

Multiple conditions – Test the value of multiple conditions to determine the correct set of steps.

Use a decision table instead of nested if-then-else Structured English constructs to simplify the presentation of complex logic that involves combinations of conditions.

A decision table is a tabular presentation of complex logic in which rows represent conditions and possible actions and columns indicate which combinations of conditions result in specific actions.

DECISION TABLE	Rule	Rule	Rule	Rule
[Condition]	value	value	value	value
[Condition]	value	value	value	value
[Condition]	value	value	value	value
[Sequence of steps or conditional steps]	X			
[Sequence of steps or conditional steps]		X	X	
[Sequence of steps or conditional steps]				X

Although it isn't a Structured English construct, a decision table can be named, and referenced within a Structured English procedure.

One-to-many iteration – Repeat the set of steps until the condition is false.

Use this construct if the set of steps must be performed at least once, regardless of the condition's initial value.

Repeat the following until [truth condition]:
    [sequence of steps or conditional steps]
~~End Repeat~~

Zero-to-many iteration – Repeat the set of steps until the condition is false.

Use this construct if the set of steps is conditional based on the condition's initial value.

Do while [truth condition]:
    [sequence of steps or conditional steps]
~~End Do~~

- OR -

For [truth condition]:
    [sequence of steps or conditional steps]
~~End For~~

Policies and Decision Tables

Policy – a set of rules that govern show a process is to be completed.

Decision table – a tabular form of presentation that specifies a set of conditions and their corresponding actions.

– As required to implement a policy.

A Simple Decision Table

A SIMPLE POLICY STATEMENT

CHECK CASHING IDENTIFICATION CARD

A customer with check cashing privileges is entitled to cash personal checks of up to $75.00 and payroll checks from companies pre-approved by *LMART*. This card is issued in accordance with the terms and conditions of the application and is subject to change without notice. This card is the property of *LMART* and shall be forfeited upon request of *LMART*.

SIGNATURE *Charles C. Parker, Jr.*

EXPIRES **May 31, 2003**

THE EQUIVALENT POLICY DECISION TABLE

Conditions and Actions	Rule 1	Rule 2	Rule 3	Rule 4
C1: Type of check	personal	payroll	personal	payroll
C2: Check amount less than or equal to $75.00	yes	doesn't matter	no	doesn't matter
C3: Company accredited by *LMART*	doesn't matter	yes	doesn't matter	no
A1: Cash the check	X	X		
A2: Don't cash the check			X	X

Condition Stubs

Action Stubs

Rules

Data Flows & Control Flows

Data flow – data that is input to or output from a process.

– A data flow is data in motion

Data flow name →

– A data flow may also be used to represent the creation, reading, deletion, or updating of data in a file or database (called a data store).

Composite data flow – a data flow that consists of other data flows.
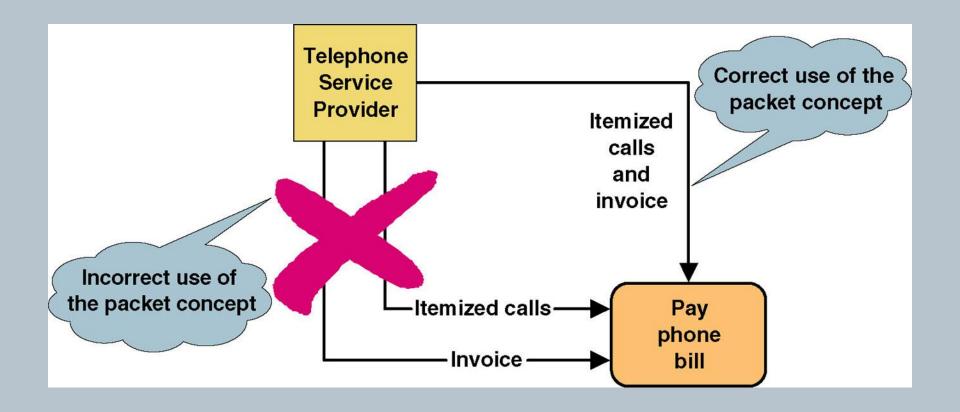
Control flow – a condition or nondata event that triggers a process.

Control flow name ⊳

– Used sparingly on DFDs.
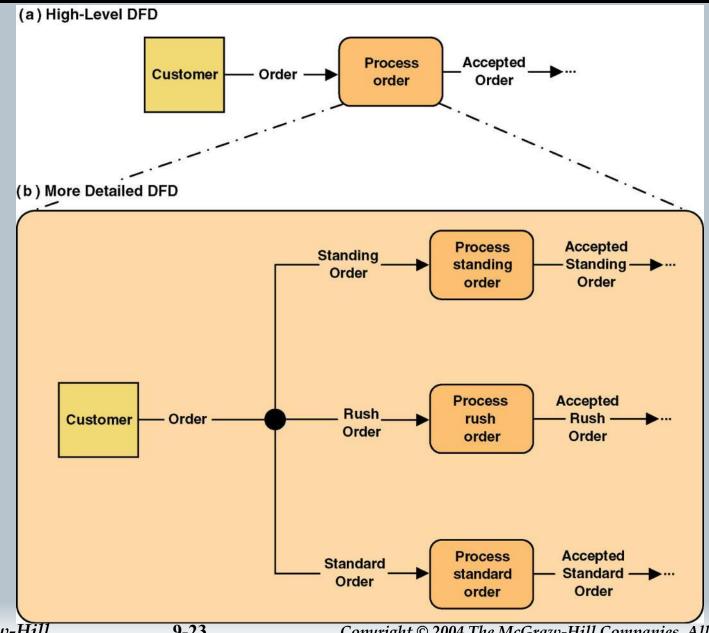
Data Flow Packet Concept
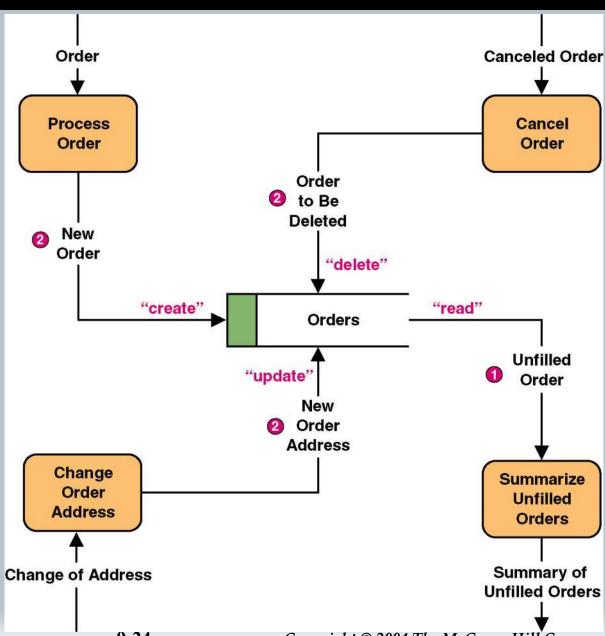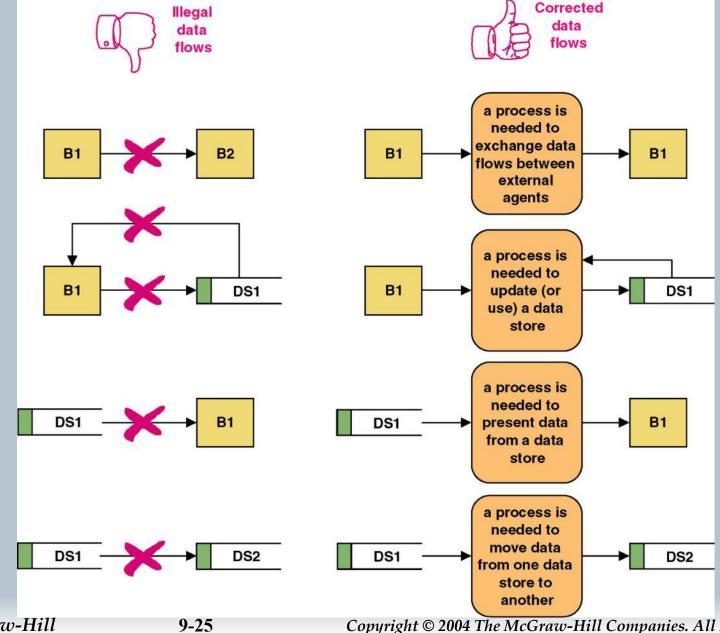
Composite and Elementary Data Flows

(a) High-Level DFD

(b) More Detailed DFD

Data Flows to and from Data Stores

Illegal Data Flows

Data Conservation

Data conservation – the practice of ensuring that a data flow contains only data needed by the receiving process.

- Sometimes called *starving the processes*.
- New emphasis on business process redesign to identify and eliminate inefficiencies.
- Simplifies the interface between those processes.
- Must precisely define the data composition of each data flow, expressed in the form of *data structures*.

Data Structures

Data attribute – the smallest piece of data that has meaning to the users and the business.

Data structure – a specific arrangement of data attributes that defines a single instance of a data flow..

– The data attributes that comprise a data flow are organized into data structures.

– Data flows can be described in terms of the following types of data structures:

- A *sequence* or group of data attributes that occur one after another.
- The *selection* of one or more attributes from a set of attributes.
- The *repetition* of one or more attributes.

A Data Structure for a Data Flow

DATA STRUCTURE	ENGLISH INTERPRETATION

ORDER=
    ORDER NUMBER +
    ORDER DATE+
    [PERSONAL CUSTOMER NUMBER,
     CORPORATE ACCOUNT NUMBER]+
    SHIPPING ADDRESS=ADDRESS+
    (BILLING ADDRESS=ADDRESS)+
    1 {PRODUCT NUMBER+
      PRODUCT DESCRIPTION+
      QUANTITY ORDERED+
      PRODUCT PRICE+
      PRODUCT PRICE SOURCE+
      EXTENDED PRICE } N+
    SUM OF EXTENDED PRICES+
    PREPAID AMOUNT+
    (CREDIT CARD NUMBER+EXPIRATION
    DATE)
    (QUOTE NUMBER)
 	
ADDRESS=
    (POST OFFICE BOX NUMBER)+
    STREET ADDRESS+
    CITY+
    [STATE, MUNICIPALITY]+
    (COUNTRY)+
    POSTAL CODE
	An instance of ORDER consists of:
	    ORDER NUMBER and
	    ORDER DATE and
	    Either PERSONAL CUSTOMER NUMBER
	     or CORPORATE ACCOUNT NUMBER
	    and SHIPPING ADDRESS (which is
	    equivalent to ADDRESS)
	    and optionally: BILLING ADDRESS (which is
	     equivalent to ADDRESS)
	    and one or more instances of:
	     PRODUCT NUMBER and
	     PRODUCT DESCRIPTION and
	     QUANTITY ORDERED and
	     PRODUCT PRICE and
	     PRODUCT PRICE SOURCE and
	     EXTENDED PRICE
	    and SUM OF EXTENDED PRICES and
	    PREPAID AMOUNT and
	    optionally: both CREDIT CARD NUMBER
	    and EXPIRATION DATE
	
	An instance of ADDRESS consists of:
	    optionally: POST OFFICE BOX NUMBER and
	    STREET ADDRESS and
	    CITY and
	    Either STATE or MUNICIPALITY
	    and optionally: COUNTRY
	    and POSTAL CODE

Data Structure Constructs

Data Structure	Format by Example (relevant portion is **boldfaced**	English Interpretation (relevant portion is **boldfaced**)
Sequence of Attributes - The sequence data structure indicates one or more attributes that may (or must) be included in a data flow.	WAGE AND TAX STATEMENT= **TAXPAYER IDENTIFICATION NUMBER+ TAXPAYER NAME+ TAXPAYER ADDRESS+ WAGES, TIPS, AND COMPENSATION+ FEDERAL TAX WITHHELD+...**	An instance of WAGE AND TAX STATEMENTS consists of: **TAXPAYER IDENTIFICATION NUMBER and TAXPAYER NAME and TAXPAYER ADDRESS and WAGES, TIPS AND COMPENSATION and FEDERAL TAX WITHHELD and...**
Selection of Attributes - The selection data structure allows you to show situations where different sets of attributes describe different instances of the data flow.	ORDER= **(PERSONAL CUSTOMER NUMBER, CORPORATE ACCOUNT NUMBER)+** ORDER DATE+...	An instance or ORDER consists of: **Either PERSONAL CUSTOMER NUMBER or CORPORATE ACCOUNT NUMBER;** and ORDER DATE and...

Data Structure Constructs (continued)

Data Structure	Format by Example (relevant portion is **boldfaced**	English Interpretation (relevant portion is **boldfaced**)

Repetition of Attributes - The repetition data structure is used to set off a data attribute or group of data attributes that may (or must) repeat themselves a specific number of time for a single instance of the data flow.
  The minimum number of repetitions is usually *zero* or *one*.
  The maximum number of repetitions may be specified as "n" meaning "many" where the actual number of instances varies for each instance of the data flow.
	POLICY NUMBER+
	  POLICYHOLDER NAME+
	  POLICY HOLDER ADDRESS+
	  0 {DEPENDENT NAME+
	    DEPENDENT'S
	RELATIONSHIP} N+
	  1 {EXPENSE DESCRIPTION+
	    SERVICE PROVIDER+
	    EXPENSE AMOUNT} N
		An instance of CLAIM consists of:
		  POLICY NUMBER and
		  POLICYHOLDER NAME and
		  POLICYHOLDER ADDRESS and
		  zero or more instance of:
		    DEPENDENT NAME and
		    DEPENDENT'S
		RELATIONSHIP and
		  one or more instances of:
		    EXPENSE DESCRIPTION and
		    SERVICE PROVIDER and
		    EXPENSE ACCOUNT

Data Structure Constructs (concluded)

Data Structure	Format by Example (relevant portion is **boldfaced**	English Interpretation (relevant portion is **boldfaced**)

Optional Attributes - The optional notation indicates that an attribute, or group of attributes <u>in a sequence or selection date structure</u> may not be included in all instances of a data flow.
Note: For the repetition data structure, a minimum of "zero" is the same as making the entire repeating group "optional."
	CLAIM=
	   POLICY NUMBER+
	   POLICYHOLDER NAME+
	   POLICYHOLDER ADDRESS+
	   (SPOUSE NAME+
	      DATE OF BIRTH)+…
		An instance of CLAIM consists of:
		   POLICY NUMBER and
		   POLICYHOLDER NAME and
		   POLICYHOLDER ADDRESS and
		   optionally, SPOUSE NAME and
		   DATE OF BIRTH and…

Reusable Attributes - For groups of attributes that are contained in many data flows, it is desirable to create a separate data structure that can be reused in other data structures.	DATE=
	   MONTH+
	   DAY+
	   YEAR+
		Then, the reusable structures can be included in other data flow structures as follows:
		   ORDER=ORDER NUMBER…+DATE
		   INVOICE=INVOICE NUMBER…+DATE
		   PAYMENT=CUSTOMER NUMBER…+DATE

Data Types and Domains

Data attributes should be defined by data types and domains.

Data type - a class of data that be stored in an attribute.
– Character, integers, real numbers, dates, pictures, etc.

Domain – the legitimate values for an attribute.

Diverging and Converging Data Flows
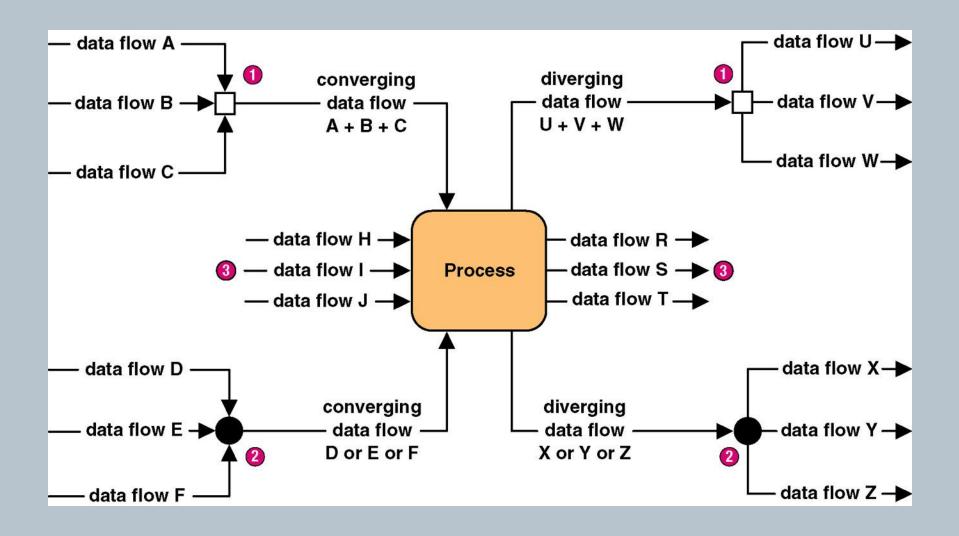
Diverging data flow – a data flow that splits into multiple data flows.

– Indicates data that starts out naturally as one flow, but is routed to different destinations.

– Also useful to indicate multiple copies of the same output going to different destinations.

Converging data flow – the merger of multiple data flows into a single packet.

– Indicates data from multiple sources that can (must) come together as a single packet for subsequent processing.

Diverging and Converging Data Flows

External Agents

External agent – an outside person, organization unit, system, or organization that interacts with a system. Also called an *external entity*.

– External agents define the "boundary" or scope of a system being modeled.

– As scope changes, external agents can become processes, and vice versa.

– Almost always one of the following:

- Office, department, division.
- An external organization or agency.
- Another business or another information system.
- One of your system's end-users or managers

– Named with descriptive, singular noun

External Agent

Gane and Sarson shape

External Agent

DeMarco/Yourdon shape

Data Stores

Data store – stored data intended for later use. Synonyms are *file* and *database*.

– Frequently implemented as a file or database.

– A data store is "data at rest" compared to a data flow that is "data in motion."

– Almost always one of the following:
  - Persons (or groups of persons)
  - Places
  - Objects
  - Events (about which data is captured)
  - Concepts (about which data is important)

– Data stores depicted on a DFD store all instances of data entities (depicted on an ERD)

– Named with plural noun

	Data Store

Gane and Sarson shape

Data Store

DeMarco/Yourdon shape

When to Draw Process Models

- Strategic systems planning
  - Enterprise process models illustrate important business functions.
- Business process redesign
  - "As is" process models facilitate critical analysis.
  - "To be" process models facilitate improvement.
- Systems analysis (primary focus of this course)
  - Model the existing system including its limitations
  - Model the target system's logical requirements (meaning processes and data flows needed regardless of how the system will be implemented)
  - Model candidate technical solutions (physical DFDs only)
  - Model the target technical solution (physical DFDs only)
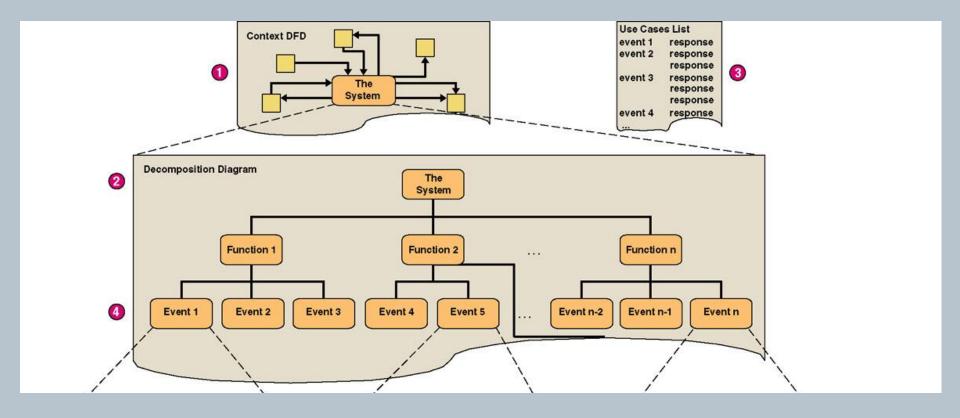
Classical Structured Analysis

1. Draw top-down physical DFDs that represent the current physical implementation of the system including its limitations.

2. Convert the physical DFDs to their logical equivalents.

3. Draw top-down logical DFDs that represent an improved system.

4. Describe all data flows, data stores, policies, and procedures in a data dictionary or encyclopedia.

5. Optionally, mark up copies of the logical DFDs to represent alternative physical solutions.

6. Draw top-down physical DFDs that represent the target solution.
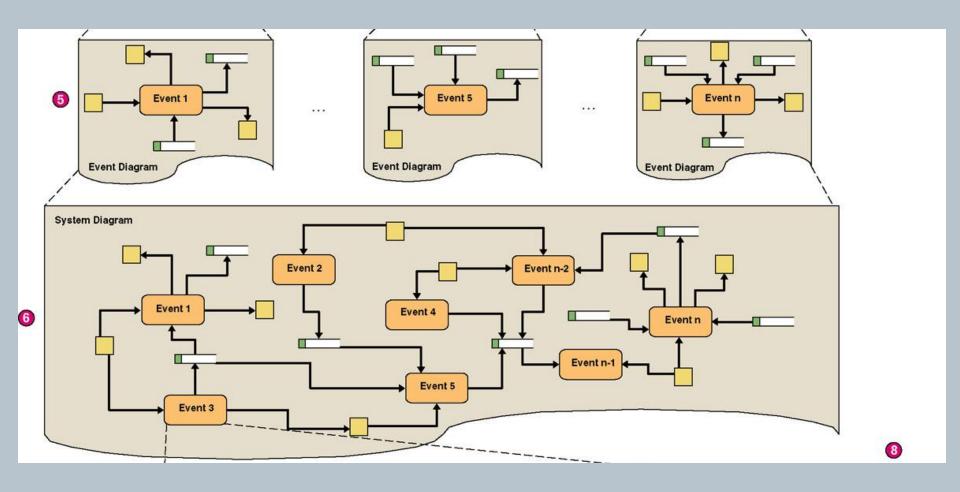
   THE ABOVE METHODOLOGY IS RARELY PRACTICED ANYMORE BECAUSE IT IS VERY CUMBERSOME AND TIME-CONSUMING.

Modern Structured Analysis

1. Draw a context DFD to establish initial project scope.
2. Draw a functional decomposition diagram to partition the system into subsystems.
3. Create an event-response or use-case list for the system to define events for which the system must have a response.
4. Draw an event DFD (or event handler) for each event.
5. Merge event DFDs into a system diagram (or, for larger systems, subsystem diagrams).
6. Draw detailed, primitive DFDs for the more complex event handlers.
7. Document data flows and processes in the data dictionary.
    THE ABOVE METHODOLOGY, BASED ON EVENT PARTITIONING, IS MORE COMMONLY PRACTICED.
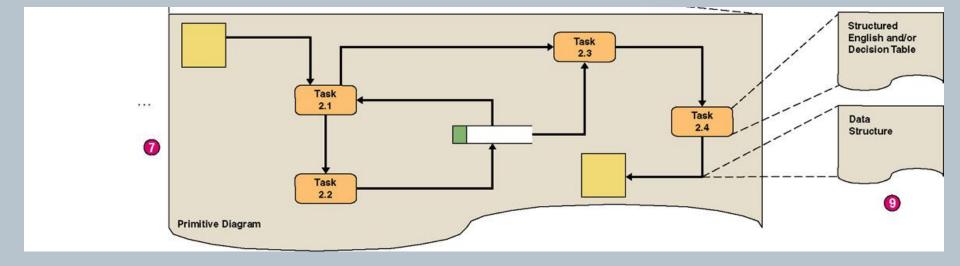
Structured Analysis Diagram Progression (1 of 3)
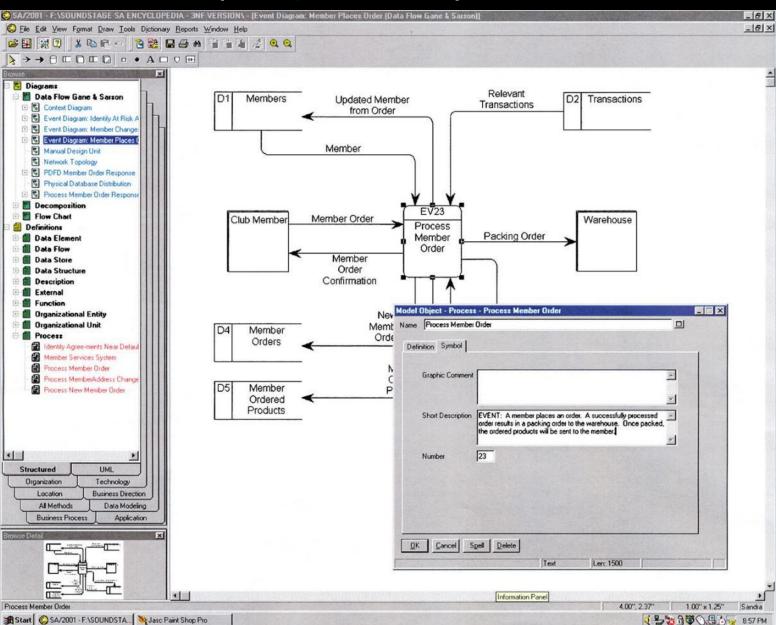
Structured Analysis Diagram Progression   (2 of 3)
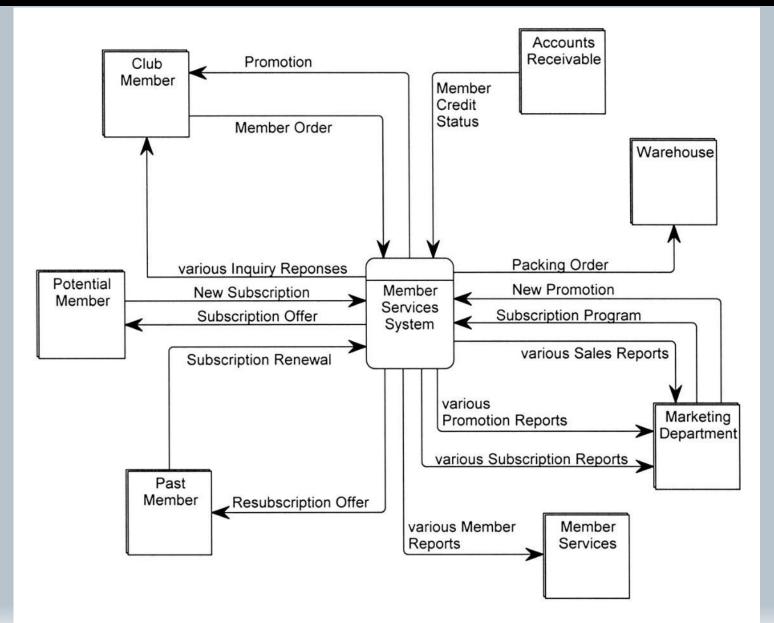
Structured Analysis Diagram Progression   (3 of 3)
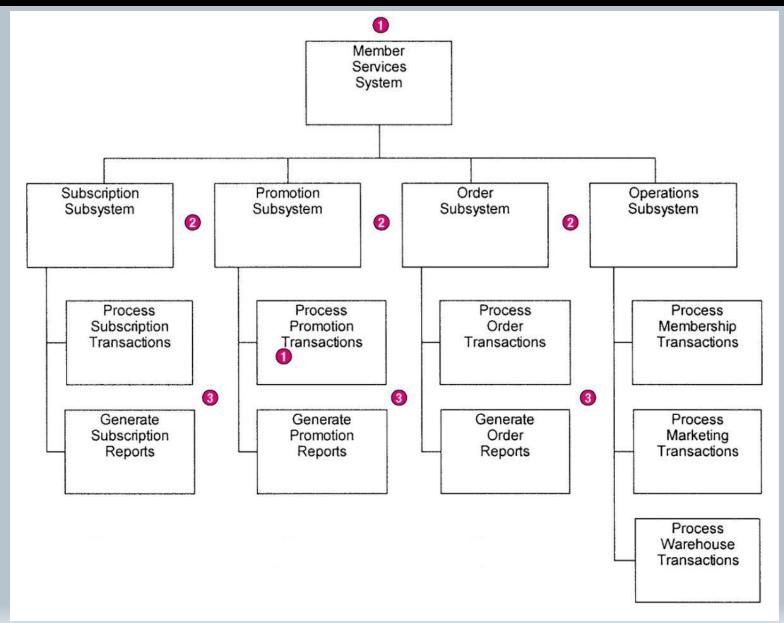
CASE for DFDs (Sample Screen) from System Architect 2001

SoundStage Context DFD

SoundStage Functional Decomposition Diagram

Events

- **External events** are initiated by external agents. They result in an input transaction or data flow.

- **Temporal events** are triggered on the basis of time, or something that merely happens. They are indicated by a control flow.

- **State events** trigger processes based on a system's change from one state or condition to another. They are indicated by a control flow.

Use Cases

Use case – an analysis tool for finding and identifying business events and responses.

Actor – anything that interacts with a system.
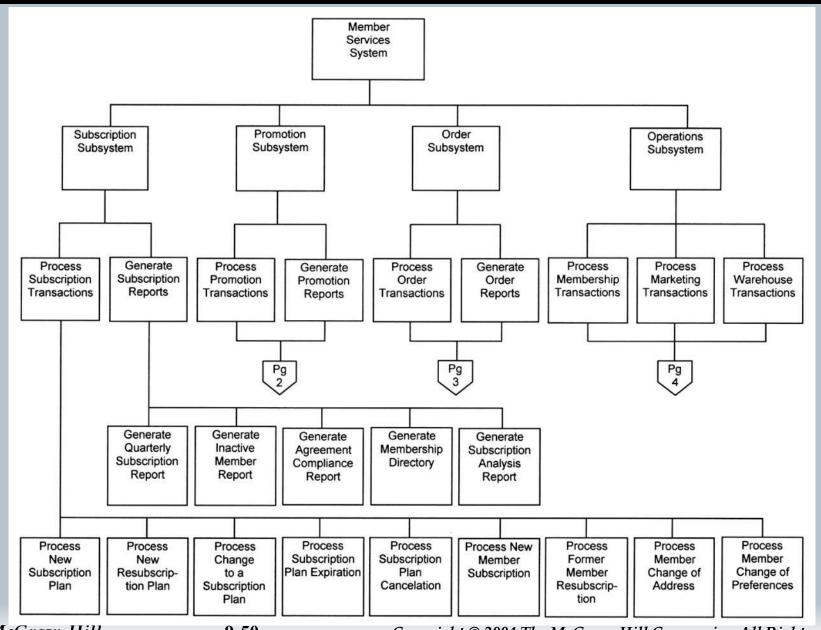
Use Case List

Actor	Event (or Use Case)	Trigger	Responses
Marketing	Establishes a new membership subscription plan to entice new members.	NEW MEMBER SUBSCRIPTION PROGRAM	Generate SUBSCRIPTION PLAN CONFIRMATION. Create AGREEMENT in the database.
Marketing	Establishes a new membership resubscription plan to lure back former members.	PAST MEMBER RESUBSCRIPTION PROGRAM	Generate SUBSCRIPTION PLAN CONFIRMATION. Create AGREEMENT in the database.
Marketing	Changes a subscription plan for current members (e.g., extending the fulfillment period).	SUBSCRIPTION PLAN CHANGE	Generate AGREEMENT CHANGE CONFIRMATION. Update AGREEMENT in the database.
(time)	A subscription plan expires.	(current date)	Generate AGREEMENT CHANGE CONFIRMATION. Logically Delete (void) AGREEMENT in the database.
Marketing	Cancels a subscription plan before its planned expiration date.	SUBSCRIPTION PLAN CANCELLATION	Generate AGREEMENT CHANGE CONFIRMATION. Logically Delete (void) AGREEMENT in the database.
Member	Joins the club by subscribing. ("Take any 12 CDs for one penny and agree to buy 4 more at regular prices within two years.")	NEW SUBSCRIPTION	Generate MEMBER DIRECTORY UPDATE CONFIRMATION. Create MEMBER in the database. Create first MEMBER ORDER and MEMBER ORDERED PRODUCTS in the database.
Member	Changes address (including e-mail and privacy code).	CHANGE OF ADDRESS	Generate MEMBER DIRECTORY UPDATE CONFIRMATION. Update MEMBER in the database.
Accounts Receivable	Changes member's credit status.	CHANGE OF CREDIT STATUS	Generate CREDIT DIRECTORY UPDATE CONFIRMATION. Update MEMBER in the database.
(time)	90 days after Marketing decides to no longer sell a product.	(current date)	Generate CATALOG CHANGE CONFIRMATION. Logically Delete (deactivate) PRODUCT in the database.
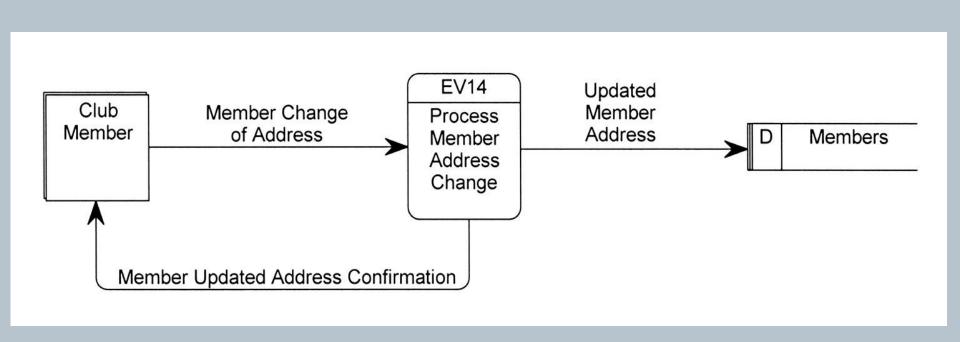
Use Case List (continued)

Actor	Event (or Use Case)	Trigger	Responses
Member	Wants to pick products for possible purchase. (Logical requirement is driven by vision of web-based access to information.)	PRODUCT INQUIRY	Generate CATALOG DESCRIPTION.
Member	Places order.	NEW MEMBER ORDER	Generate MEMBER ORDER CONFIRMATION. Create MEMBER ORDER and MEMBER ORDERED PRODUCTS in the database.
Member	Revises order.	MEMBER ORDER CHANGE REQUEST	Generate MEMBER ORDER CONFIRMATION. Update MEMBER ORDER and/or MEMBER ORDERED PRODUCTS in the database.
Member	Cancels order.	MEMBER ORDER CANCELLATION	Generate MEMBER ORDER CONFIRMATION. Logically Delete MEMBER ORDER and MEMBER ORDERED PRODUCTS in the database.
(time)	90 days after the order.	(current date)	Physically Delete MEMBER ORDER and MEMBER ORDERED PRODUCTS in the database.
Member	Inquires about his or her purchase history (three-year time limit).	MEMBER PURCHASE INQUIRY	Generate MEMBER PURCHASE HISTORY.
(each) Club	(end of month).	(current date)	Generate MONTHLY SALES ANALYSIS. Generate MONTHLY MEMBER AGREEMENT EXCEPTION ANALYSIS. Generate MEMBERSHIP ANALYSIS REPORT.
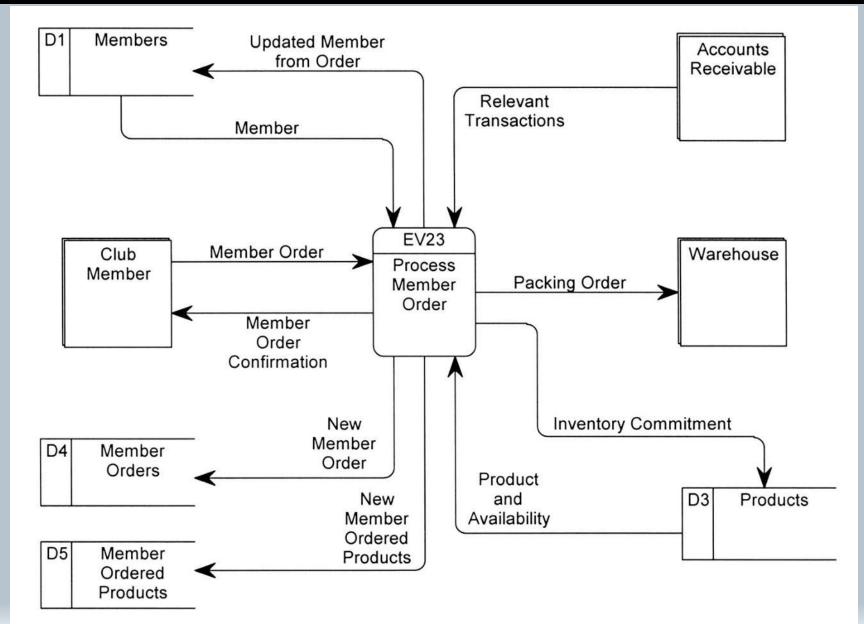
Event Decomposition Diagram (partial)

External Event DFD
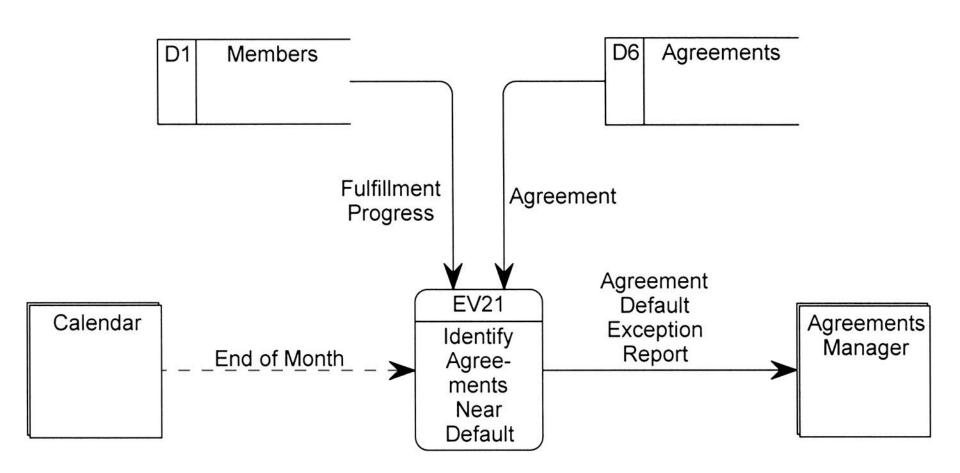
Event diagram – data flow diagram that depicts the context for a single event.
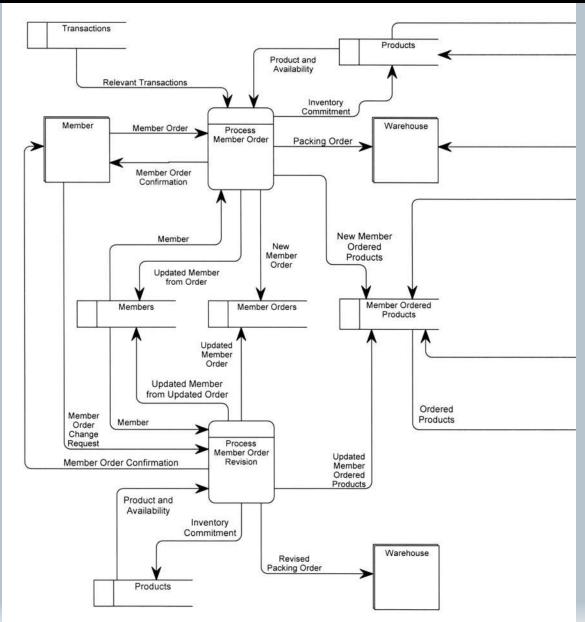
External Event DFD (more complex)

Temporal Event DFD
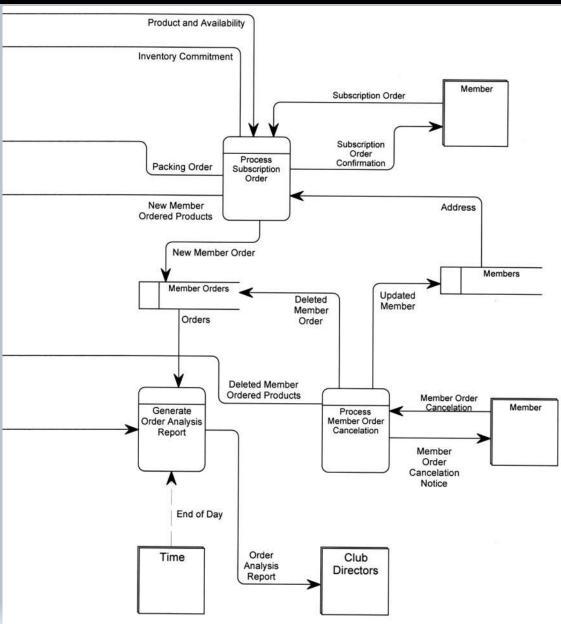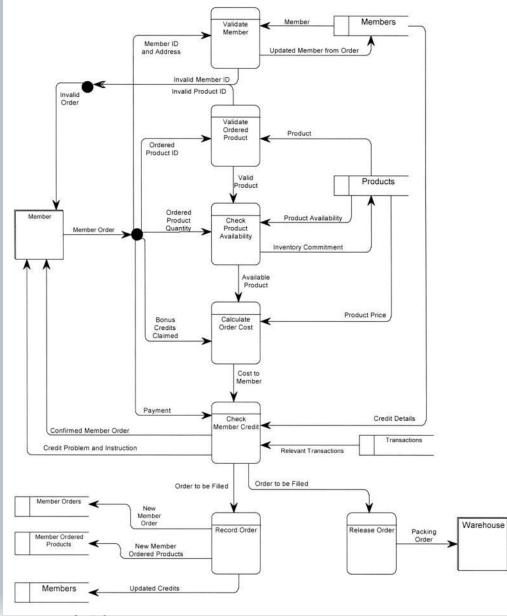
System DFD
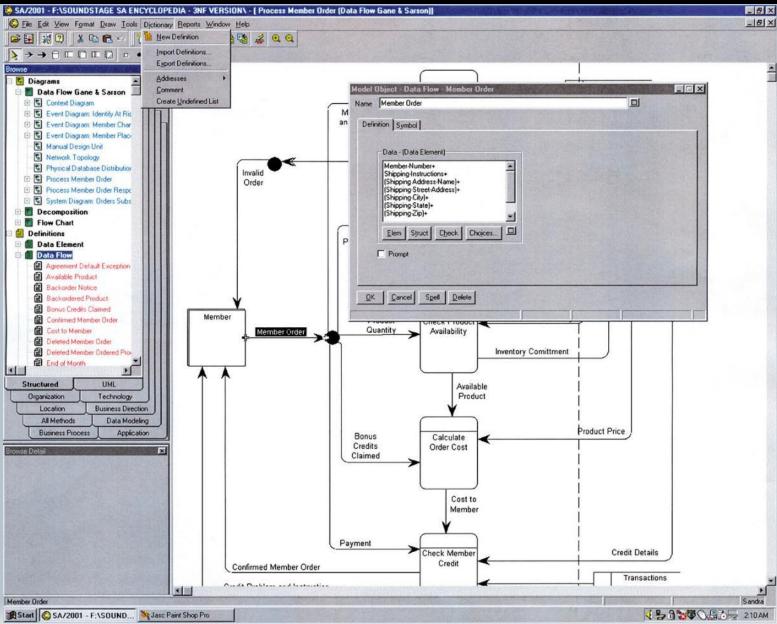
System DFD
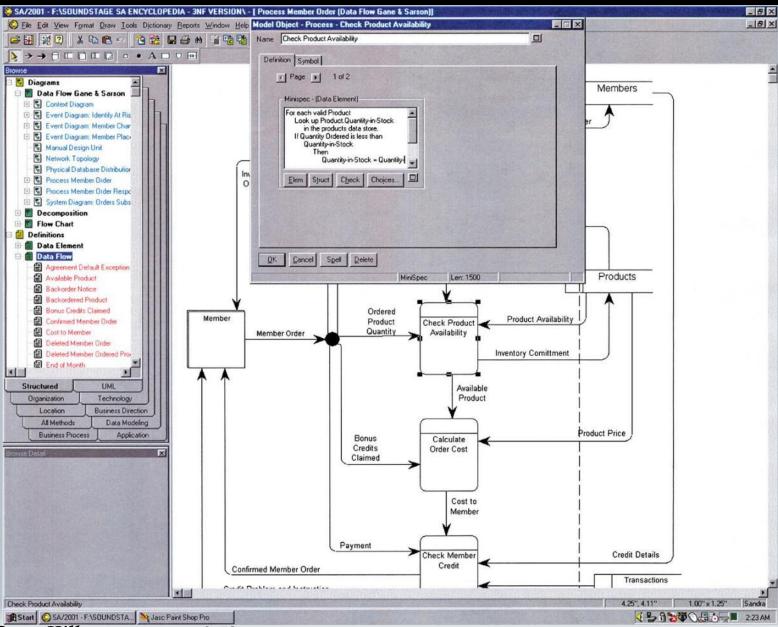
Primitive DFD (see book for more readable copy)

Entering a Data Flow Using a CASE Tool

Entering an Elementary Process Using a CASE Tool

Sample Data to Process CRUD Matrix

Data-to-Process-CRUD Matrix

Entity . Attribute	Process Customer Application	Process Customer Credit Application	Process Customer Change of Address	Process Internal Customer Credit Change	Process New Customer Order	Process Customer Order Cancellation	Process Customer Change to Outstanding Order	Process Internal Change to Customer Order	Process New Product Addition	Process Product Withdrawal from Market	Process Product Price Change	Process Change to Product Specification	Process Product Inventory Adjustment
Customer	C	C			R	R	R	R					
.Customer Number	C	C			R	R	R	R					
.Customer Name	C	C	U		R		R	R					
.Customer Address	C	C	U		RU		RU	RU					
.Customer Credit Rating		C		U	R		R	R					
.Customer Balance Due					RU	U	R	R					
Order					C	D	RU	RU					
.Order Number					C		R	R					
.Order Date					C		U	U					
.Order Amount					C		U	U					
Ordered Product					C	D	CRUD	CRUD		RU			
.Quantity Ordered					C		CRUD	CRUD					
.Ordered Item Unit Price					C		CRUD	CRUD					
Product					R	R	R	R	C	D	RU	RU	RU
.Product Number					R	R	R	R	C			R	
.Product Name					R		R	R	C			RU	
.Product Description					R		R	R	C			RU	
.Product Unit of Measure					R		R	R	C		RU	RU	
.Product Current Unit Price					R		R	R			U		
.Product Quantity on Hand					RU	U	RU	RU					RU

C = create   R = read   U = update   D = delete

Sample Process to Location Association Matrix

Process-to-Location-Association Matrix

Process	Customers	Kansas City	Marketing	Advertising	Warehouse	Sales	Accounts Receivable	Boston	Sales	Warehouse	San Francisco	Sales	San Diego	Warehouse
Process Customer Application	X					X			X			X		
Process Customer Credit Application	X						X							
Process Customer Change of Address	X					X			X			X		
Process Internal Customer Credit Change							X							
Process New Customer Order	X					X			X			X		
Process Customer Order Cancellation	X					X			X			X		
Process Customer Change to Outstanding Order	X					X			X			X		
Process Internal Change to Customer Order						X			X			X		
Process New Product Addition			X											
Process Product Withdrawal from Market			X											
Process Product Price Change			X											
Process Change to Product Specification			X	X										
Process Product Inventory Adjustment					X					X				X