



**HRM/MAN 679 Management Information Systems**

**Louis Coraggio Ph.D., Associate Professor**

**Troy State University**

**Systems Analysis Class Notes “A”**

Copyright 1998  
Louis Coraggio  
All Rights Reserved

-----> BASIC TOOLS OF STRUCTURED ANALYSIS <-----

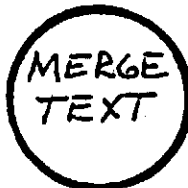
- DATA FLOW DIAGRAMS - SHOW WHERE INFO COMES FROM, WHERE IT GOES, HOW IT GETS THERE
- DATA DICTIONARY - CLEARLY DEFINE ALL TERMS
- STRUCTURED ENGLISH - WRITE SPECIFICATIONS AND DEPICT LOGIC IN AN UNDERSTANDABLE FORM
- SYSTEM HIEARCHY CHARTS - SHOW SEQUENCE OF EVENTS AND WHO'S IN CHARGE

---

DATA FLOW DIAGRAMS = A GRAPHIC REPRESENTATION OF INFORMATION FLOW THROUGH THE SYSTEM. (AKA "BUBBLE CHARTS") ONLY FOUR BASIC SYMBOLS WILL BE USED:



TERMINATOR -- INFORMATION ENTERS OR LEAVES THE SYSTEM (SOURCE OR SINK)



TRANSFORM -- INFORMATION IS CHANGED BY SOME PROCESS



FILE -- INFORMATION IS STORED FOR FUTURE USE. MAY BE ANY KIND OF MEDIA



FLOW VECTOR - A DIRECTIONAL LINE WITH A DATA STRUCTURE ATTACHED.

----> ANALYST SPEAK <----

- DATA ELEMENT ≡ SMALLEST INCREMENT OF INFORMATION WE CAN COMFORTABLY DEFINE

E.G. "ZIP CODE"

- DATA STRUCTURE ≡ COMBINATION OF DATA ELEMENTS AND STRUCTURES REFERRED TO BY ONE NAME

E.G. THE STRUCTURE "ADDRESS" COULD CONSIST OF THE ELEMENTS: "STREET", "CITY", "STATE" AND "ZIP CODE"

- DATA ≡ GENERIC SHORTHAND FOR DATA STRUCTURE OR ELEMENT AKA "INFORMATION"

- SOURCE ≡ AN EXTERNAL ENTITY WHICH INJECTS DATA INTO THE SYSTEM

E.G. "VENDOR"

- SINK ≡ AN EXTERNAL ENTITY THAT RECEIVES DATA FROM THE SYSTEM

E.G. "CONTROLLER"

- TRANSFORM ≡ PROCESS ≡ AN OPERATION THAT MERGES, SORTS, OR CHANGES DATA. MUST HAVE AT LEAST 1 ELEMENT ENTERING AND 1 NEW ELEMENT LEAVING.

E.G. "ISSUE PAYCHECKS"

DATA DICTIONARY = A SET OF DEFINITIONS OF THE DATA ELEMENTS AND STRUCTURES DEPICTED IN THE DATA FLOW DIAGRAMS.

- ESSENCE OF DEFINITION REQUIRES "GENUS" AND "DIFFERENTIA"
  - GENUS IS A CLASSIFICATION
  - DIFFERENTIA IS WHAT SETS THE OBJECT APART FROM OTHERS IN THE CLASS
  - GENUS SHOULD HAVE MULTIPLE DIFFERENTIA

■ EXAMPLE CLASSIFICATION SCHEME

<u>GENUS</u>	&	<u>DIFFERENTIA</u>
ANIMAL		MAMMAL, IVERTIBRATES, ...
MAMMAL		CANINE, RODENT, PRIMATE...
CANINE		DOG, WOLF, FOX ...
DOG		COCKER SPANIEL, PUG, ...

- FOR OUR PURPOSES WE WILL CLASSIFY THREE TYPES OF INFO IN OUR DATA DICTIONARY:
  - DATA FLOW - USUALLY A DATA STRUCTURE THAT SHOWS UP ON A FLOW VECTOR
  - FILES - WHERE DATA FLOWS PAUSE TO REST
  - DATA ELEMENTS - INDIVISIBLE DATA FLOW

STRUCTURED ENGLISH ≡ A SUBSET OF THE ENGLISH LANGUAGE WHICH EXPLAINS HOW DATA ELEMENTS AND STRUCTURES ARE TRANSFORMED WITHIN A PROCESS.

(AKA PSEUDO-CODE)

IN STRUCTURED ENGLISH WE WILL:

- TRY TO THINK AS STUPIDLY AS A COMPUTER
  - STRICTLY LIMIT THE
    - VOCABULARY;
    - SYNTAX OF STATEMENTS; AND
    - WAYS TO COMBINE STATEMENTS
  - TRY TO ELIMINATE THE NUANCES AND CONTEXT PROBLEMS INHERENT IN THE LANGUAGE ("CREDIT")
  - USE THIS TECHNIQUE TO WRITE LOGICAL AND PERFORMANCE SPECIFICATIONS THAT ARE INDEPENDENT OF AUTOMATION
- 

SYSTEM HIERARCHY CHART ≡ AN "ORGANIZATION CHART" OF THE INFORMATION SYSTEM SHOWING LEVELS OF AUTHORITY, WHO REPORTS TO WHOM, AND DIRECT ASSOCIATIONS.

- IMPORTANT FOR MAINTENANCE TO FIGURE OUT HOW WIDESPREAD ANY CHANGE IS
- ESTABLISH A PRIORITY FOR DEVELOPMENT AND TESTING
- FIGURE OUT WHERE THE SYSTEM GOES FOR INFORMATION AND WHERE IT SHOULD BE DELIVERED

--> PROCEDURE FOR STRUCTURED ANALYSIS AND DESIGN <--

«» TO ATTACK THE INCREDIBLY COMPLEX PROBLEM OF SYSTEM DESIGN WE NEED SOME WAY SIMPLIFY THE PROBLEM AND BREAK IT DOWN INTO MANAGEABLE PIECES.

### STRUCTURED ANALYSIS HIERARCHY

1. IDENTIFY INFORMATION SOURCES AND SINKS
  2. DEFINE THE DATA ELEMENTS AND STRUCTURES ASSOCIATED WITH SOURCES AND SINKS
  3. IDENTIFY AND THE FLOW OF DATA AS IT PASSES THROUGH THE SYSTEM, NOTING PROCESSES THAT TRANSFORM DATA
  4. DECOMPOSE THE TRANSFORMATIONS INTO SETS OF SMALLER PROCESSES. CONTINUE UNTIL THE PROCESSES CAN BE MADE NO SMALLER.
  5. SPECIFY EACH TRANSFORM IN TERMS OF:
    - WHAT GOES IN
    - WHAT GOES OUT
    - WHAT MUST HAPPEN INSIDE
  6. LOOK AT EACH TRANSFORM SPECIFICATION AND DETERMINE THE LOGIC FOR DOING IT
- «» ALL OF THIS TO BE DONE WHILE STRICTLY ADHERING TO OUR PRINCIPLES!

DATA DICTIONARY ENTRIES FOR FORM LETTER SYSTEM  
(PLAIN OLD ENGLISH)

CUSTOMER ADDRESS	STRUCTURE CONTAINING: STREET NUMBER AND NAME, CITY, STATE AND ZIP CODE
CUSTOMER NAME	STRUCTURE CONTAINING FIRST AND LAST NAME WITH TITLES
CUSTOMER NAME & ADDRESS	STRUCTURE CONTAINING <u>CUSTOMER NAME</u> AND <u>CUSTOMER ADDRESS</u>
FORM LETTER	STRUCTURE CONTAINING <u>TEXT OF LETTER</u> AND <u>CUSTOMER NAME &amp; ADDRESS</u>
INVALID CUSTOMER LIST	STRUCTURE CONTAINING ALL <u>CUSTOMER</u> <u>NAMES</u> THAT APPEAR ON <u>SELECTED</u> <u>CUSTOMER NAMES</u> BUT ARE NOT IN THE <u>MASTER MAILING LIST</u>
MASTER MAILING LIST	STORE CONTAINING ALL CURRENT <u>CUSTOMER NAME &amp; ADDRESS</u> STRUCTURES
SELECTED CUSTOMER NAMES	STRUCTURE CONTAINING ALL <u>CUSTOMER</u> <u>NAMES</u> TO BE USED FOR THIS MAILING
TEXT OF LETTER	ELEMENT CONSISTING OF RANDOM LENGTH TEXT

STRUCTURED ENGLISH PROCEDURE FOR FORM LETTER SYSTEM  
(SOME WHAT SIMPLIFIED)

FOR EACH SELECTED CUSTOMER NAME:

CHECK CUSTOMER NAME AGAINST MASTER MAILING LIST

IF CUSTOMER NAME IS ON MASTER MAILING LIST;

PUT CUSTOMER NAME & ADDRESS WITH

TEXT OF LETTER TO COMPLETE FORM LETTER.

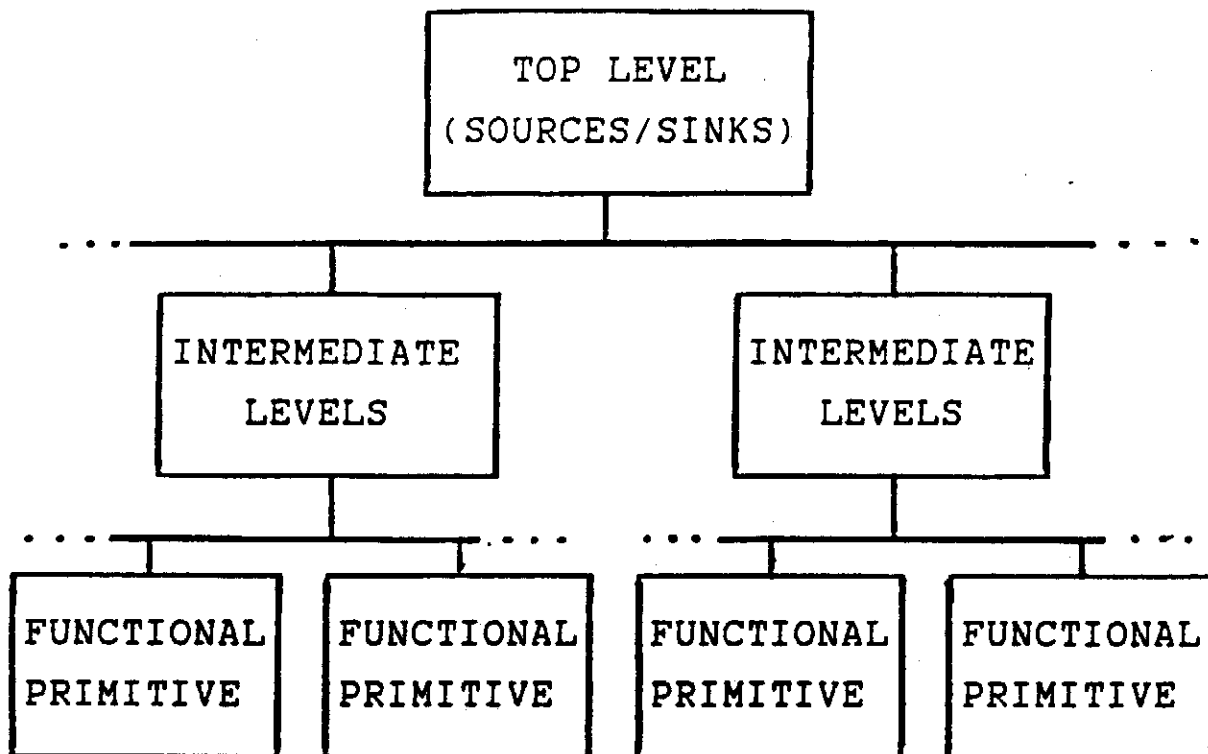
OTHERWISE

PUT CUSTOMER NAME ON INVALID CUSTOMER LIST.

MAIL ALL FORM LETTERS TO CUSTOMERS.

SEND INVALID CUSTOMER LIST TO MARKETING.

## TYPICAL STRUCTURE FOR ANALYSIS (HIERARCHY CHART)



### ADVANTAGES TO DECOMPOSITION

- ALLOWS FOR TOP DOWN ANALYSIS AND TOP DOWN SPECIFICATIONS
- CAN SHOW ONLY THOSE DETAILS APPROPRIATE FOR THE LEVEL IN QUESTION
- NO OFF-PAGE CONNECTORS
- CAN BE EASILY REPRODUCED
- FOLLOWS A LOGICAL METHOD OF SUBDIVIDING THE ANALYSTS TASKS

-----> SOME RULES AND GUIDELINES FOR <-----  
BUILDING DATA FLOW DIAGRAMS

RULES

1. DATAFLOWS AND TERMINATORS SHOULD BE NOUN PHRASES
  - Blanket\_Vendor
  - Validated\_Vendor\_Invoice
2. TRANSFORMS SHOULD BE VERB PHRASES AND DENOTE THE ACTIONS TAKEN
  - PROCESS-TIME-CARDS
  - VERIFY-ACCOUNT-BALANCE
3. ALL DATAFLOWS SHOULD BE CONSERVED WITHIN A TRANSFORM OR FILE/STORE. IF IT GOES IN IT'S GOT TO COME OUT
  - IF IT'S A PROCESS AND EVERY DATAFLOW COMES IN, IT'S REALLY A SINK
  - IF IT'S A PROCESS AND EVERY DATAFLOW LEAVES, IT'S REALLY A SOURCE
4. NO DATA FLOWS ARE PERMITTED FROM ONE TERMINATOR DIRECTLY TO ANOTHER TERMINATOR
5. SOMETHING MUST HAPPEN IN EVERY TRANSFORM. IF NOTHING HAPPENS, IT ISN'T A PROCESS
  - MERGE: TWO OR MORE DATAFLOWS ARE COMBINED
  - SORT: A DATAFLOW IS DIVIDED
6. THE SYSTEM HAS TO BE SELF CONTAINED
  - ALL INFORMATION MUST COME FROM A SOURCE
  - ALL INFORMATION SHOULD GO TO A SINK

-----> WHAT IS DECOMPOSITION? <-----

«» DECOMPOSE MEANS BREAKING DOWN SOMETHING INTO ITS  
FUNDAMENTAL COMPONENTS

«» KNOWN BY MANY NAMES, "SUBDIVIDING" "LEVELING"

BASICALLY A WAY TO OVERCOME LIMITATIONS OF DRAWING AND  
PAGE SIZES. ALSO A STRUCTURED METHOD FOR BREAKING DOWN  
A COMPLEX PROBLEM INTO MANAGEABLE SIZE.

EACH TRANSFORM CAN BE DECOMPOSED INTO ANOTHER DFD WHICH  
PROVIDES MORE DETAIL

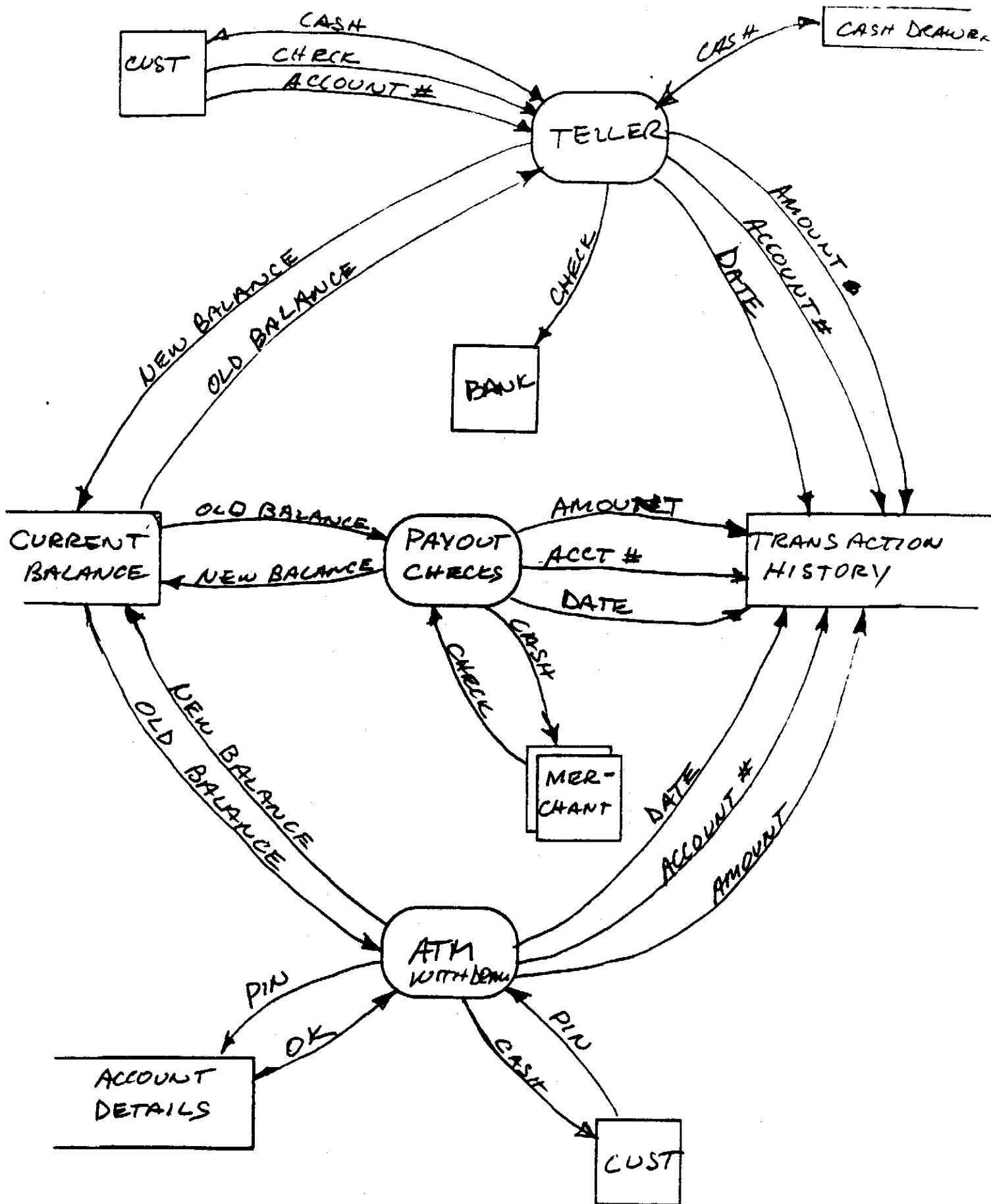
LIKE THE REST OF OUR TECHNIQUES, DECOMPOSITION IS BASED  
ON PRINCIPLES:

- PARENT-CHILD RELATIONSHIPS: ALL OF THE CHILDREN  
MUST BE INCLUDED IN THE PARENT PROCESS. THE CHILD  
PROCESS MUST ADDRESS ALL INPUTS AND OUTPUTS OF THE  
PARENT.
- FUNCTIONAL PRIMITIVES: THE SMALLEST "NUCLEAR"  
PROCESS WE CAN IDENTIFY THRU DECOMPOSITION. WE  
WILL KNOW A FUNCTIONAL PRIMITIVE BECAUSE IT:
  - HAS NO INTERNAL DATAFLOWS;
  - HAS A SINGLE INPUT AND SINGLE OUTPUT; OR
  - IS TRIVIAALLY SIMPLE.

## GUIDELINES

- A. START WITH THE SOURCES AND SINKS. FIND ALL THE NET INPUTS AND OUTPUTS
- B. CLEARLY DEFINE AND NAME EACH DATAFLOW
- C. ESTABLISH MERGE AND SORT PATHS  
(YOURDAN CALLS THIS "CONNECTIVITY")
- D. WORK FROM OUTSIDE IN. A TRANSFORM WILL BE REQUIRED FOR EVERY MERGE, SORT AND CHANGE IN A DATAFLOW
- E. KEEP THE FIRST PASS SIMPLE AND DESCRIPTIVE--  
MEANINGFUL NAMES ARE MOST IMPORTANT
- F. BE PREPARED TO START OVER
- G. WHEN SATISFIED, BEGIN DECOMPOSITION

# BANK SYSTEM OVERVIEW



## A PERSPECTIVE CHECKLIST FOR ANALYSTS

- THE USER IS THE FIRST, LAST AND ONLY REASON FOR MY EXISTENCE. THE SYSTEM EXISTS TO SERVE THE USER.
- ERRORS OF OMISSION ARE A GREVOUS SIN, ERRORS OF COMMISSION CAN BE FIXED.
- MY JOB IS TO SAVE THE COMPANY MONEY AND IMPROVE SERVICE TO ITS CUSTOMERS. IF THIS WORK ISN'T FURTHERING THOSE GOALS, I PROBABLY SHOULDN'T BE DOING IT.
- THE COMPUTER IS MY SLAVE, NOT MY MASTER.
- I WILL NOT CUT A LINE OF CODE UNTIL I KNOW HOW IT FITS INTO THE SYSTEM.
- I WILL QUESTION EVERYTHING WE DO NOW -- NO MATTER HOW STUPID I LOOK.
- I WILL LOOK FOR A SIMPLER WAY.
- I WILL DECIDE ON "WHAT" BEFORE I WORRY ABOUT "HOW"
- WHEN I'M UP TO MY BUTT IN ALLIGATORS, I WILL REMEMBER I SET OUT TO DRAIN THE SWAMP.