Shortcourse on

Next Generation Database Systems

11-12 February 1988

Portland State University

Organized by
Oregon Database Forum

Sponsored by
Oregon Center for Advanced Technology Education (OCATE)

In Cooperation with
Willamette Valley ACM

OREGON DATABASE FORUM

OCATE
OREGON CENTER FOR ADVANCED TECHNOLOGY EDUCATION

Schedule

Thursday, 11 February

9:00-10:00	Introduction: Maier
10:00-10:30	Break
10:30-12:00	Carey
12:00-1:00	Lunch
1:00-2:00	Carey
2:00-3:00	Tolbert
3:00-3:30	Break
3:30-5:00	Tolbert

Friday, 12 February

9:00-10:30	Stonebraker
10:30-11:00	Break
11:00-12:00	Stonebraker
12:00-1:00	Lunch
1:00-2:30	Zdonik
2:30-3:00	Break
3:00-4:00	Zdonik
4:00-4:30	Reception
4:30-5:30	Shootout at the OCATE Corral

Speaker/Affiliation	Topic/Project	Contributing Technology
Mike Carey U. Wisconsin	Extensible Database Systems Exodus	SE: Common Interfaces
Doug Tolbert Unisys	Semantic Database Systems SIM	KR: Semantic Data Models
Mike Stonebraker UC Berkeley	Enhanced Relational Systems Postgres	AI: Rules
Stan Zdonik Brown	Object-Oriented Databases Encore/Observer	PL: ADTs & Encaps.

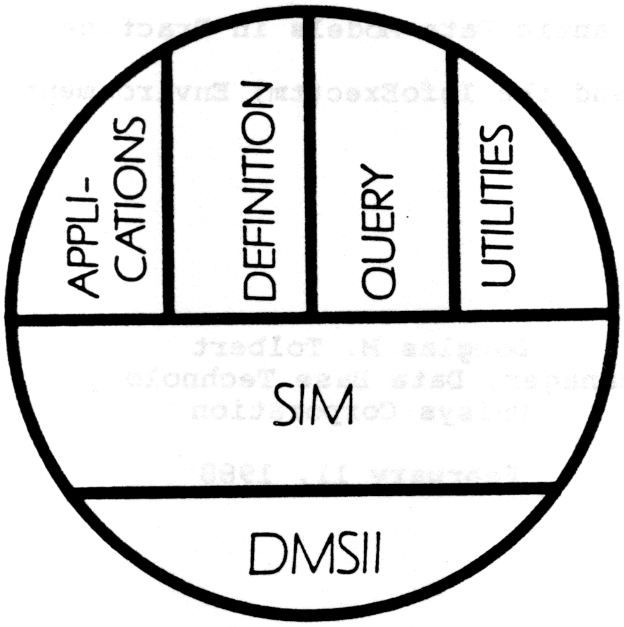
Semantic Data Models in Practice

SIM and the InfoExec(tm) Environment

Douglas M. Tolbert
Manager, Data Base Technology
Unisys Corporation

February 11, 1988

InfoExec



CONCEPTS

InfoExec Overview

Objectives

Data Models

- File Systems
- Database Systems
- Semantic Systems

SIM Concepts

An Example

- Application development effort
- Query formation

InfoExec Product Environment

SIM Architecture and Implementation

Objectives

Create an environment that minimizes effort required to manage data by providing

- Ease of Use
- Productivity
- Data Integrity
- Coexistence
- Performance

Objectives

Ease of Use

- High level interface
- Integrated interface
- Flexibility
- Data independence

Objectives

Productivity

- Set-oriented, non-procedural interface
- Naturally represent complex relationships
- Easy to learn
- Minimum programming effort
- Data retrieval without programming

Objectives

Data Integrity

- System enforced integrity
- Shared data definitions
- Referential integrity

Objectives

Coexistence

- No impact on existing DMSII data bases
- New capabilities for existing data bases

Performance

- Production level

Data Model

A way of describing real-world application systems

Formalized into four components

- Objects: Concepts to organize data
- Operators: Instructions to manipulate data
- Constraints: Rules to ensure correctness
- Methodology: Instructions for designing "good" data bases.

File Systems

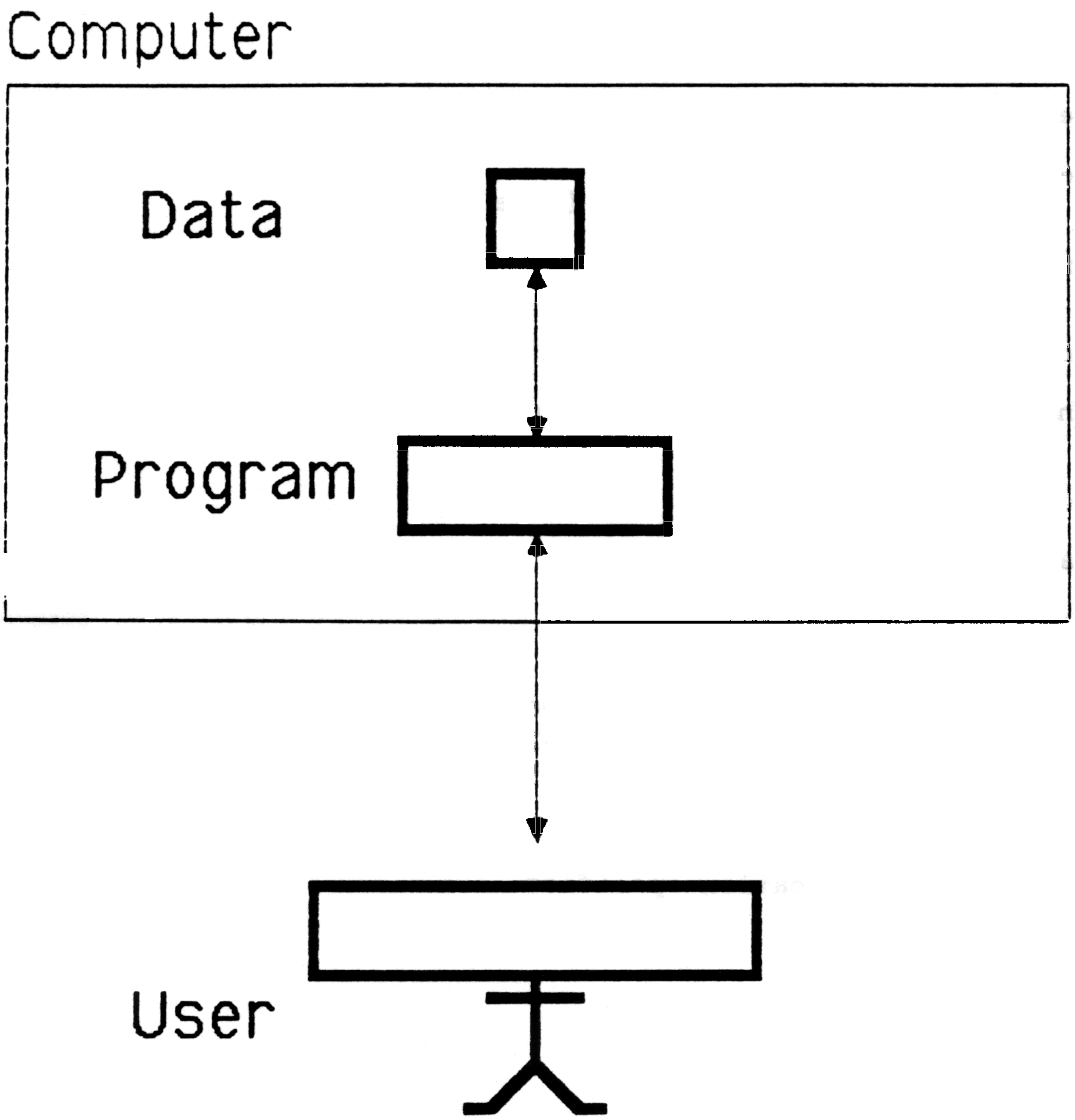
Data Model

Objects:	Records and files
Operators:	Primitive (Read and Write)
Constraints:	Almost none (parity?)
Methodology:	Good programming practices

Knowledge Location

Data:	I/O techniques
Program:	Data item integrity Simple relationships Complex relationships
User:	Complex relationships Very complex relationships Referential integrity Resource sharing (concurrency) Operational events (recovery) Application specifics

File System



Database Systems

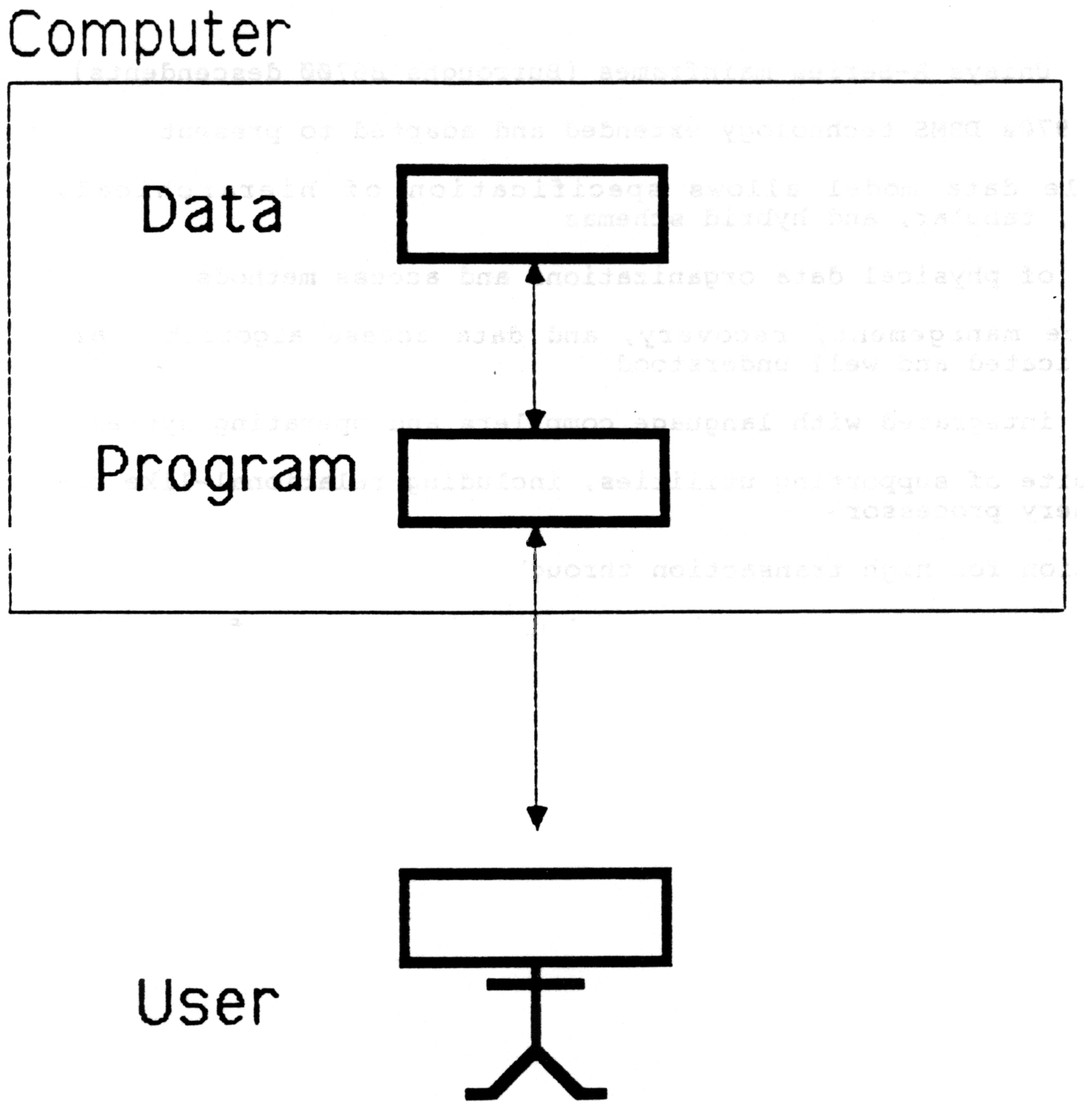
Data Model

Objects:	Records (Network) Tuples and Tables (Relational)
Operations:	Navigational (Network) Non-procedural (Relational)
Constraints:	Limited
Methodology:	Good programming practices (Network) Normalization (Relational)

Knowledge Location

Data:	Data item integrity Simple relationships (limited in Relational) Resource sharing Operational events
Program:	Complex relationships Referential integrity Application specifics
User:	Very complex relationships Referential integrity Application specifics

Database System



DMS II

Runs on Unisys A-Series mainframes (Burroughs B6700 descendents)

Early 1970s DBMS technology extended and adapted to present

Flexible data model allows specification of hierarchical network, tabular, and hybrid schemas

Variety of physical data organizations and access methods

Resource management, recovery, and data access algorithms are sophisticated and well understood

Closely integrated with language compilers and operating system

Full suite of supporting utilities, including relational-like on-line query processor

Reputation for high transaction throughput

High customer base penetration: about 95 percent of A-Series customers license DMS II

DMS II

Provides efficient basic data management functions

But data model and interfaces do not provide benefits of newer, developing technologies

Our challenge

> To provide benefits of new technology without destroying investment in existing database application systems

Options

Enhance DMS II

- Ease of Use can be improved
- Productivity and Data Integrity without sacrificing Coexistence
- Performance already near architectural limits
- Not likely to expand customer base

Build a Relational system

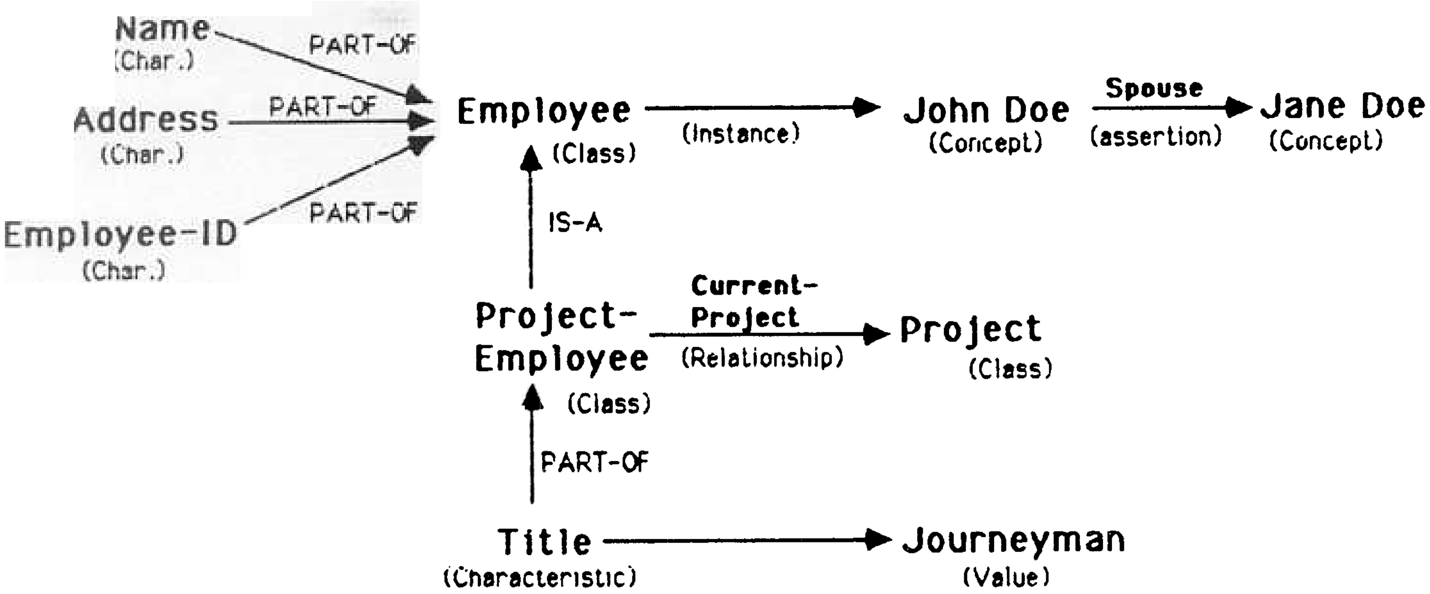
- Ease of Use would improve
- Productivity would be marginally improved
- Data Integrity very limited
- Coexistence
  - Extensive rework of basic data management software
  - No migration path for many existing DMS II data bases
  - "Two data base" strategy not desirable
- Performance questionable
- Would be one of many when released

Options

Explore new "semantic" data models

- Examined
  - Semantic Networks (Quillian, Brachman)
  - Entity-Relationship (Chen)
  - RM/T (Codd)
  - SDM (Hammer and McLeod)
  - DAPLEX (Shipman)
  - GORDAS (El-Masri)
- Meets Ease of Use, Productivity, and Data Integrity objectives
- Many new features can be adapted to existing databases without migration
- Performance based on proven DMS II algorithms
- SDM provided
  - Clear conceptual basis
  - Best DMS II coexistence possibilities
  - Good vehicle for query language development

Semantic Net



Semantic Data Model References

Abiteboul, S., and R. Hull. 1987. IFO: A formal semantic database model. ACM Trans. Database Syst. 12, 525-565.

Brachman, R. J. 1979. "On the epistemological status of semantic networks," in Associative Networks (Findler, N., ed), pp 3-50. Academic Press, New York.

Chen, P. P. 1976. The entity-relationship model: Toward a unified view of data. ACM Trans. Database Syst. 1, 9-36.

Codd, E. F. 1979. Extending the database relational model to capture more meaning. ACM Trans. Database Syst. 4, 397-434.

El-Masri, R. A. 1981. GORDAS: A Data Definition, Query and Update Language for the Entity-Category-Relationship Model of Data. Honeywell Computer Science Technical Report HR-81-250.

Hammer, M., and D. McLeod. 1981. Database description with SDM: a semantic database model. ACM Trans. Database Syst. 6, 351-386.

Hull, R., and R. King. Semantic data modeling: Survey, applications, and research issues. ACM Comput. Surv. (to appear).

Quillian, M. R. 1968. "Semantic memory," in Semantic Information Processing (Minsky, M., ed), pp 227-270. MIT Press, Cambridge, MA.

Shipman, D. W. 1981. The functional data model and the data language DAPLEX. ACM Trans. Database Syst. 6, 140-173.

Smith, J. M., and D. C. P. Smith. 1977. Database Abstractions: Aggregation and Generalization. ACM Trans. Database Syst. 2, 105-133.

Stachowitz, R. A. 1985. A formal framework for describing and classifying semantic data models. Inform. Systems 10, 77-96.

Tsichritzis, D. C., and F. H. Lochovsky. 1982. Data Models. Prentice-Hall.

Tsur, S., and C. Zaniolo. 1984. An implementation of GEM-supporting a semantic data model on a relational back-end. Proc. ACM-SIGMOD Conference.

What makes a data model semantic?

"During the last few years numerous investigations have been aimed at capturing (in a reasonably formal way) more of the meaning of the data, while preserving independence of implementation. This activity is sometimes called semantic data modeling. Actually, the task of capturing the meaning of data is a never-ending one. So the label 'semantic' must not be interpreted in any absolute sense."

Codd, pp 397-398

That is, they attempt to capture the meaning of the data, not just physical data values.

So, the more "meaning" a data model captures, the more "semantic" it is.

How does a semantic data model capture meaning?

Abstract objects for modeling data, not physical storage containers

> "... it is appropriate that the structure of a database mirror the structure of the system that it models. A database whose organization is based on naturally occuring structures will be easier for a database designer to construct and modify than one that forces him to translate the primitives of his problem domain into artificial specificaiton constructs."
>
> Hammer and McLeod, pp 351-352

Abstraction techniques

- Aggregation
  - Relationship between objects regarded as a higher level object ("PART_OF").
- Generalization
  - Collection of individual objects viewed as single object ("IS_A").

How does a semantic data model capture meaning?

Database description allows expression of

- Relationships between objects
  - one-to-one, one-to-many, many-to-many
  - generalization hierarchies
- Limits on values that attributes may take
  - type mechanism (a la Pascal)
  - subrange enforcement
  - uniqueness
- Cardinality constraints
  - required value
  - minimum and maximum instances
  - single- and multi-valued
- Ad hoc constraints
  - non-structural constraints

Semantic Systems

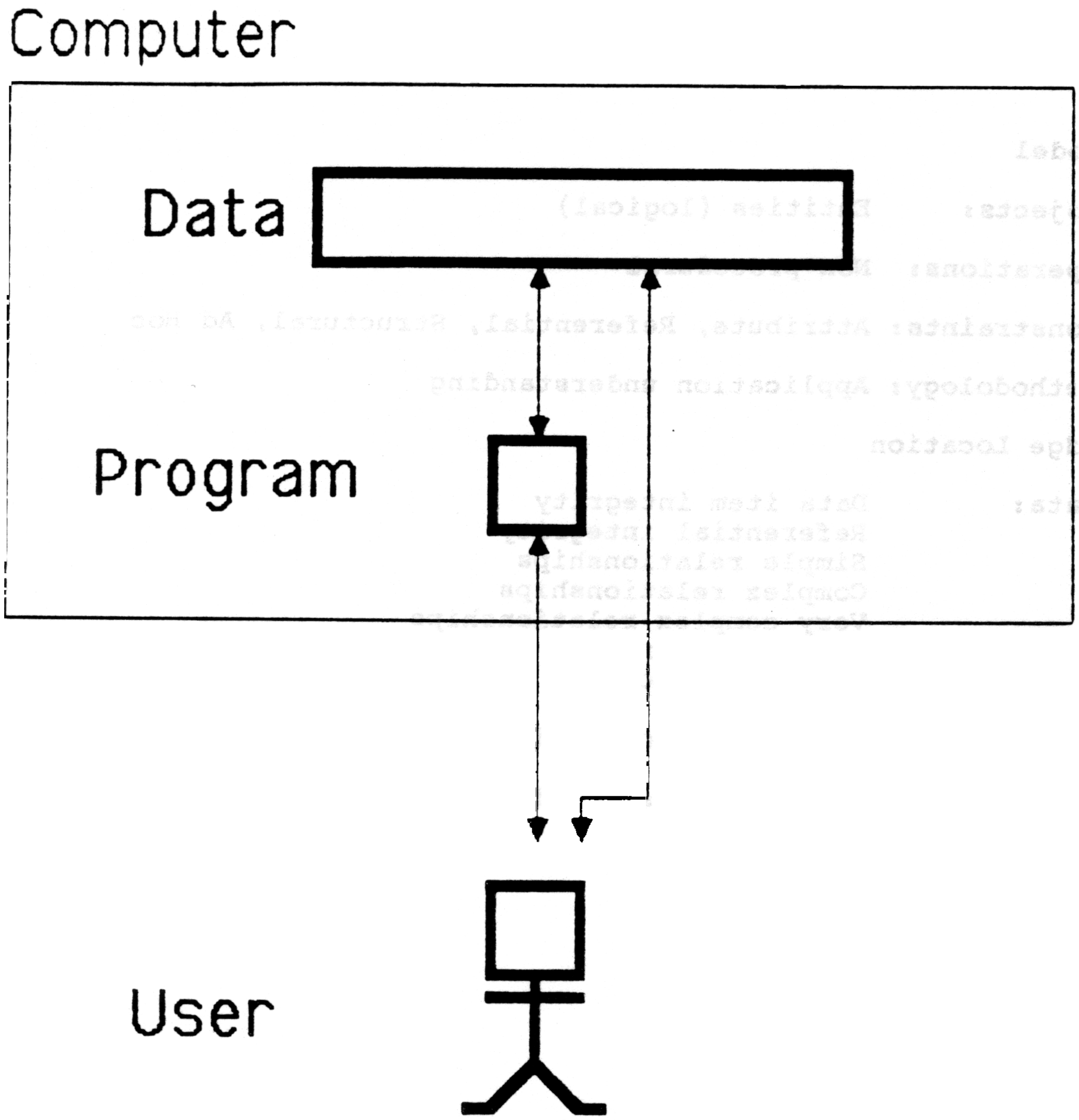
Data Model

Objects:	Entities (logical)
Operations:	Non-procedural
Constraints:	Attribute, Referential
Methodology:	Application understand

Knowledge Location

Data:	Data item integrity Referential integrity Simple relationships Complex relationships Very complex relationships Resource sharing Operational events
Program:	Application specifics
User:	Application specifics

Semantic System



SIM Concepts

Entity

An object of interest in the application environment

For example: John Doe, an employee

Accounting, a department

Annual Report Preparation, a project

SIM Concepts

Attribute

A characteristic of an entity

For example: Employee John Doe has a name, address and employee ID.

Class

A collection of entities of the same type

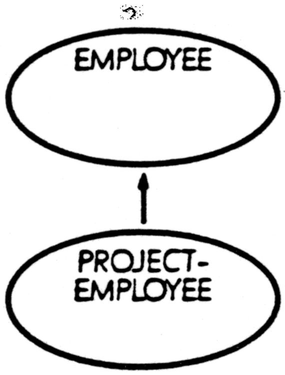
Does not imply any specific physical implementation

For example: Employee, a collection of all employees working for a company

Subclass

A subset of entities in a class

For example: Project-Employee, all employees that work on projects

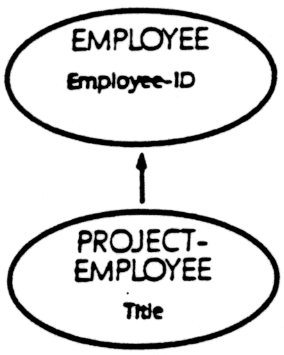


Attribute Inheritance

Subclasses inherit attributes from super classes

Additional attributes may be declared for subclasses

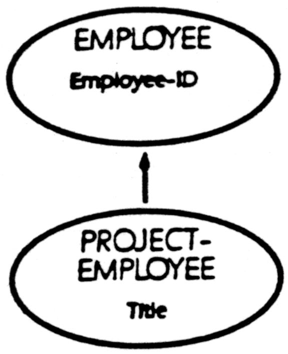
For example: Every employee has an employee ID, but only project employees have titles



Data-Valued Attributes

An attribute whose values can be displayed

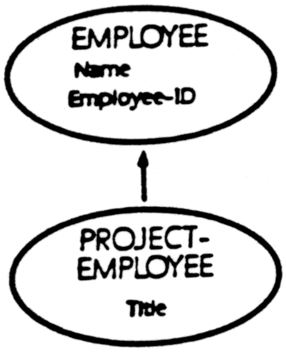
For example: An employee's ID number



Compound Attributes

A collection of attributes that may be treated as a unit

For Example: An employee's name is a compound attribute made up of first name, middle initial, and last name.

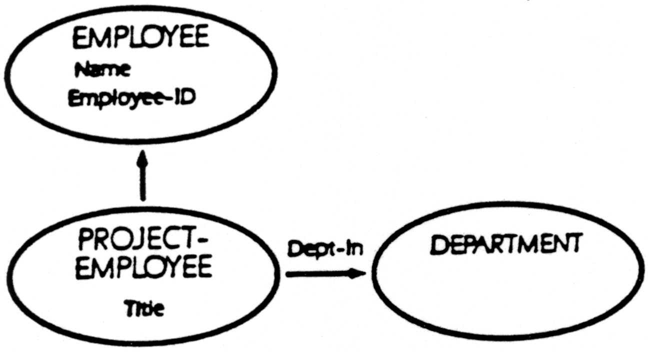


Entity-Valued Attributes

An attribute whose values are entities in a class

Indicate relationships between entities

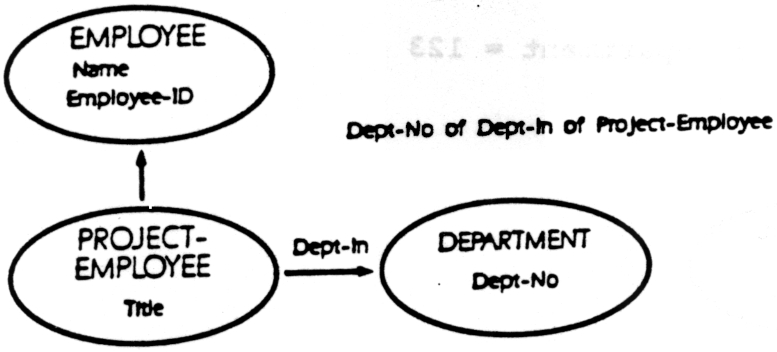
For example: A project employee's department



Extended Attributes

Attributes of related entities may be considered extended attributes of an entity

For example: The department number of a project employee



Perspective

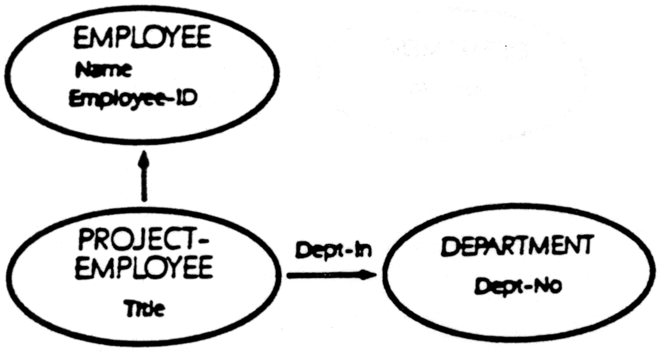
A point of view chosen for a query

Extended attributes used to view information elsewhere in the database

Same data viewed from different perspectives may have different meanings

Dept-No of Dept-In of Project-Employee = 123

Dept-No of Department = 123

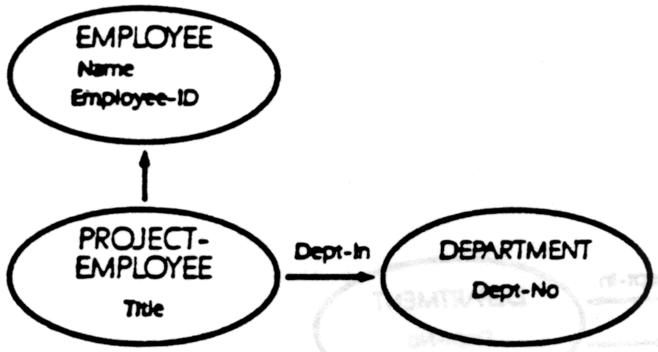


Single-Valued Attributes

An attribute that can have only one value for an entity

For example: An employee's ID number

A project employee's department

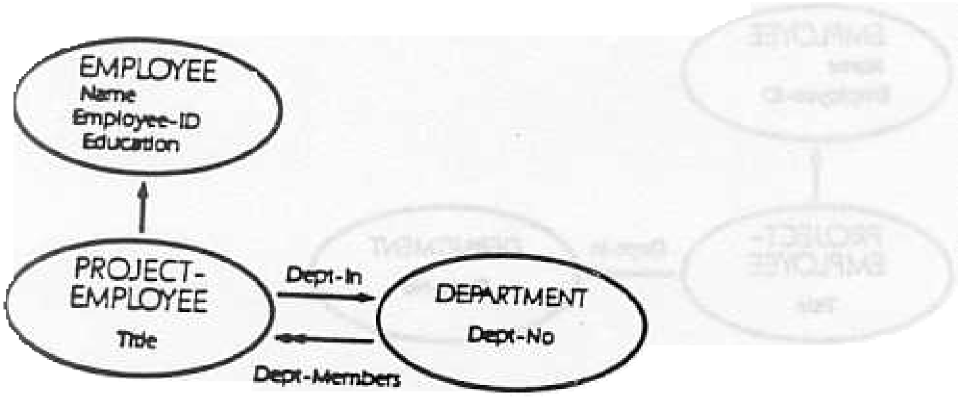


Multi-Valued Attributes

An attribute that may have more than one value for an entity

For example: An employee's education degrees

The project employees in a department

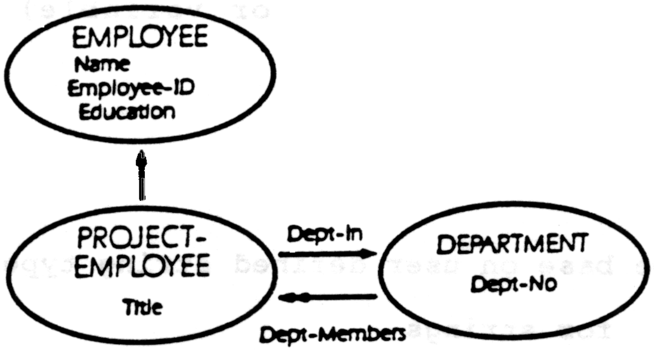


Bidirectional Relationships

Inverse relationship between two entity-valued attributes

For example: The project employees in a department, and

The members of a department



SIM Concepts

Types

Similar to Pascal

- String type checking relaxed
- More widely understood than SDM class-based domains

System Defined

- Integer, Real, Boolean, Date, Time, Character, Number (packed decimal), String (fixed or variable), Kanji, Symbolic, Ordered Symbolic

User Defined

- Based on system defined types or other user defined types
- Strings may be base on user defined string types
- Set membership for strings

Enforced subranges

- Employee-Age : Integer (18..70)

Class Attributes

Belong to class as a whole, not any particular entity

One value per class

Data-valued only

Generalized Verify

General constraints not related to schema structure

Allows user specified error messages

For example, the spouse of an employee hired after December 31, 1987 may not work for the company

```
VERIFY NoCouples ON Employee
ASSERT NOT (Spouse ISA Employee)
WHERE  Employee-Hire-Date > 12/31/87
ELSE   "Spouse may not work for company"
```

available on first release

Security

Access limits

Visibility of attributes

Retrieval and update operations

Permission associates Access with user or program

Example, the Accounting department may see but not change the salary of journeyman and higher employees

```
ACCESS LookButDontTouch ON Employee
      (Employee-Salary) RETRIEVE
WHERE Employee-Status >= Journeyman

PERMISSION
     USERCODE = Accounting,
     ACCESS = LookButDontTouch
```

Index

Visible only to database administrator and SIM Optimizer

Not visible to programmers or query users

Multiple data-valued attributes, ascending, and descending

Used for performance improvement

For Example,

```
INDEX (Employee-ID) ON Employee ASCENDING
```

ORGANIZATION -- An Example

A projects and employees data base.

Data base description

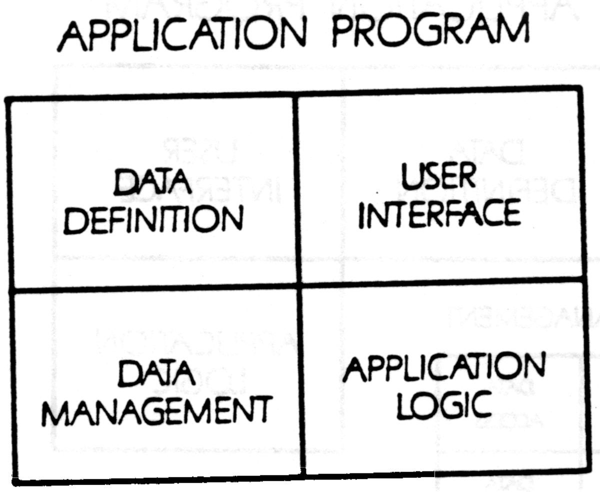
DMSII	A record oriented system
DB2	A relational system
SIM	A semantic system

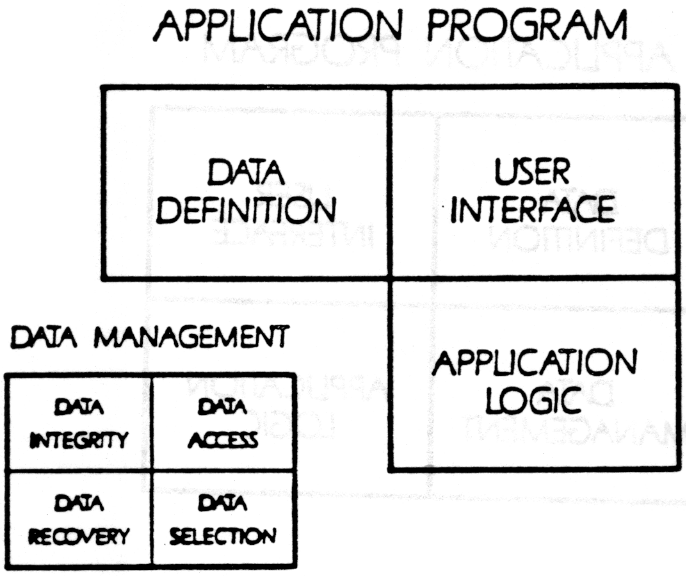
Application Responsibilities

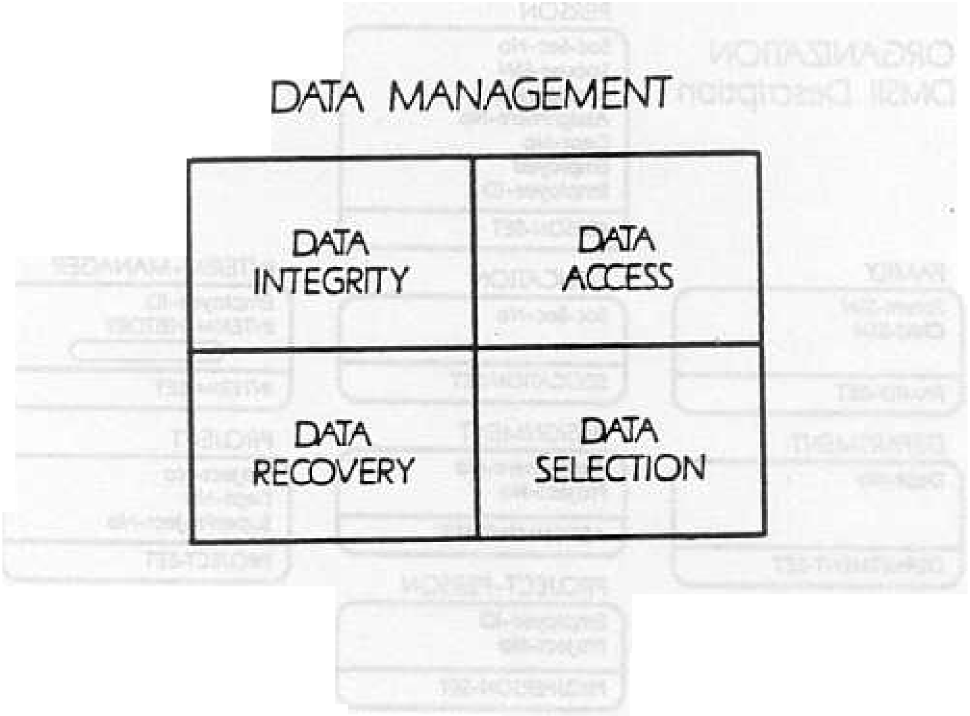
Query Formation

ORGANIZATION Entities

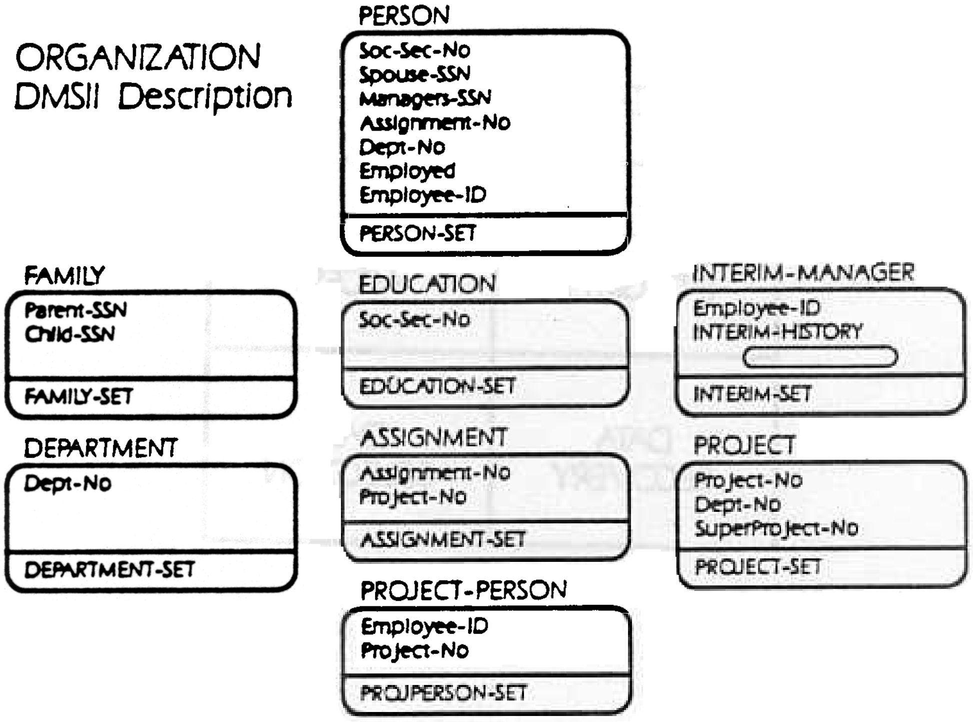
- People
- Employees
- Project Employees
- Managers
- Interim-Managers
- Projects
- Assignments
- Departments
- Previous Employees

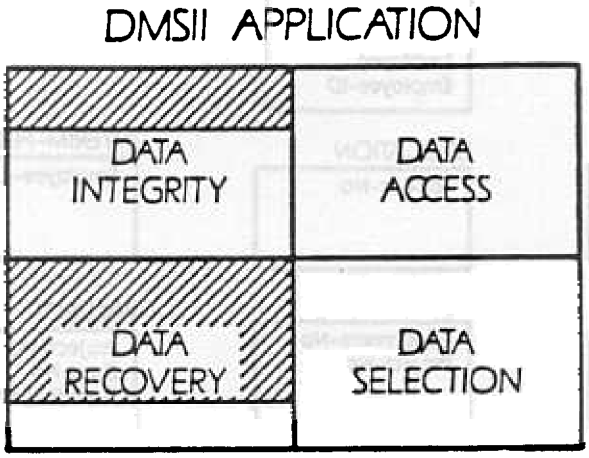

APPLICATION PROGRAM
DATA DEFINITION
USER INTERFACE
DATA MANAGEMENT
APPLICATION LOGIC

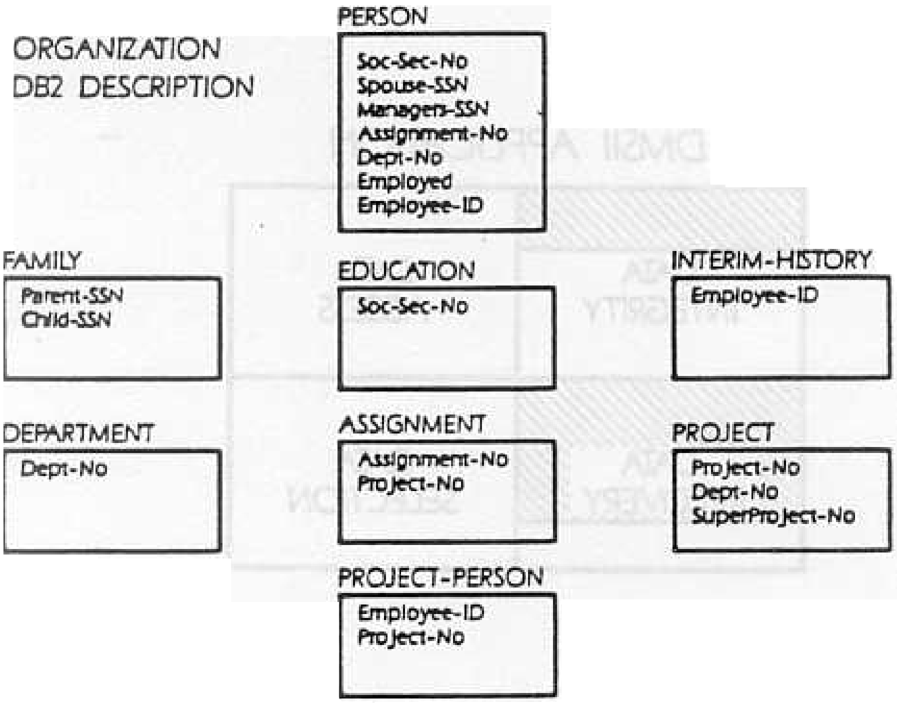

APPLICATION PROGRAM
DATA DEFINITION
USER INTERFACE
APPLICATION LOGIC
DATA MANAGEMENT
DATA INTEGRITY
DATA ACCESS
DATA RECOVERY
DATA SELECTION

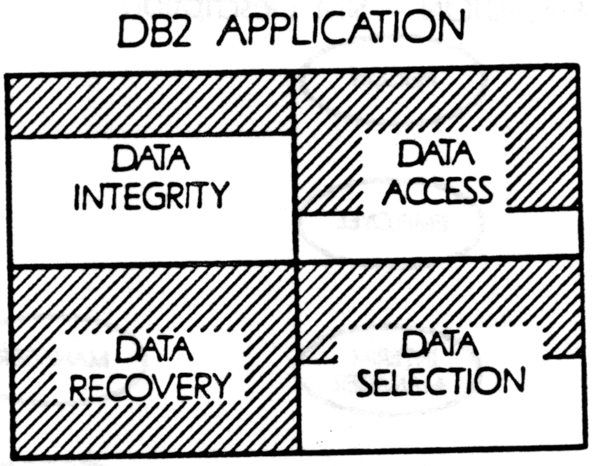

DATA MANAGEMENT
DATA INTEGRITY
DATA ACCESS
DATA RECOVERY
DATA SELECTION

ORGANIZATION DMSII Description

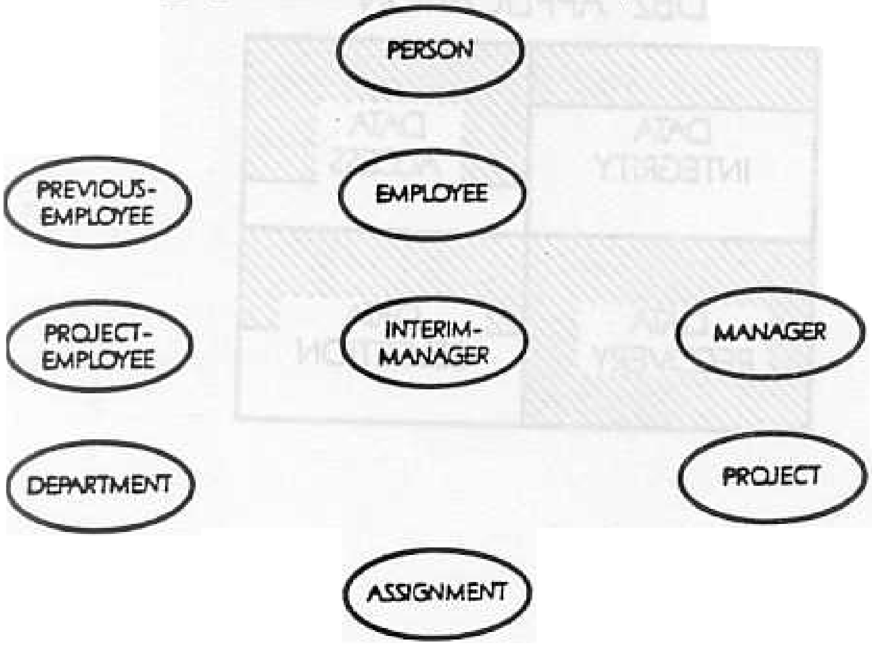



DMSII APPLICATION
DATA INTEGRITY
DATA ACCESS
DATA RECOVERY
DATA SELECTION

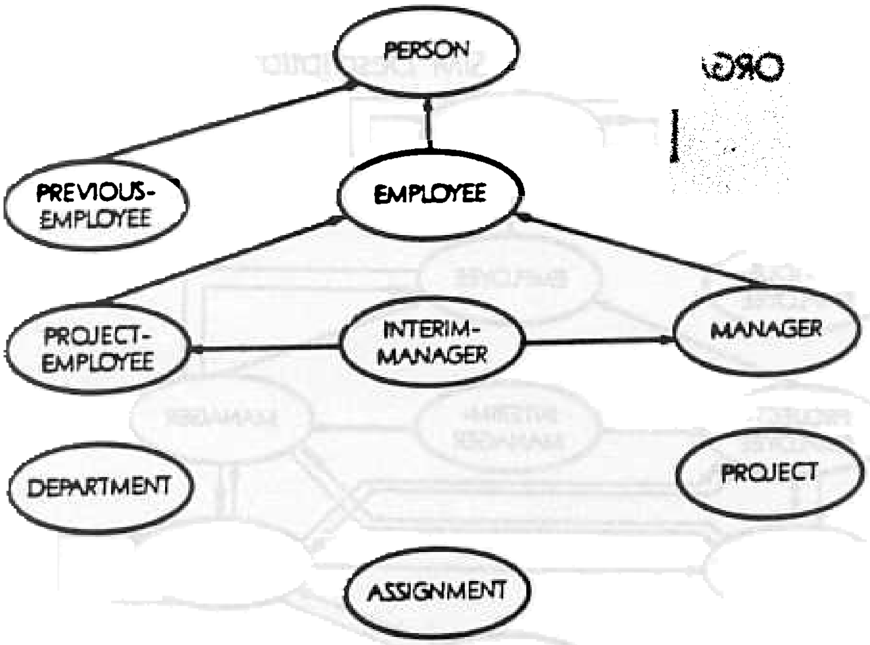

ORGANIZATION
DB2 DESCRIPTION
PERSON
Soc-Sec-No
Spouse-SSN
Managers-SSN
Assignment-No
Dept-No
Employed
Employee-ID
FAMILY
Parent-SSN
Child-SSN
EDUCATION
Soc-Sec-No
INTERIM-HISTORY
Employee-ID
DEPARTMENT
Dept-No
ASSIGNMENT
Assignment-No
Project-No
PROJECT
Project-No
Dept-No
SuperProject-No
PROJECT-PERSON
Employee-ID
Project-No


DB2 APPLICATION
DATA
INTEGRITY
DATA
ACCESS
DATA
RECOVERY
DATA
SELECTION

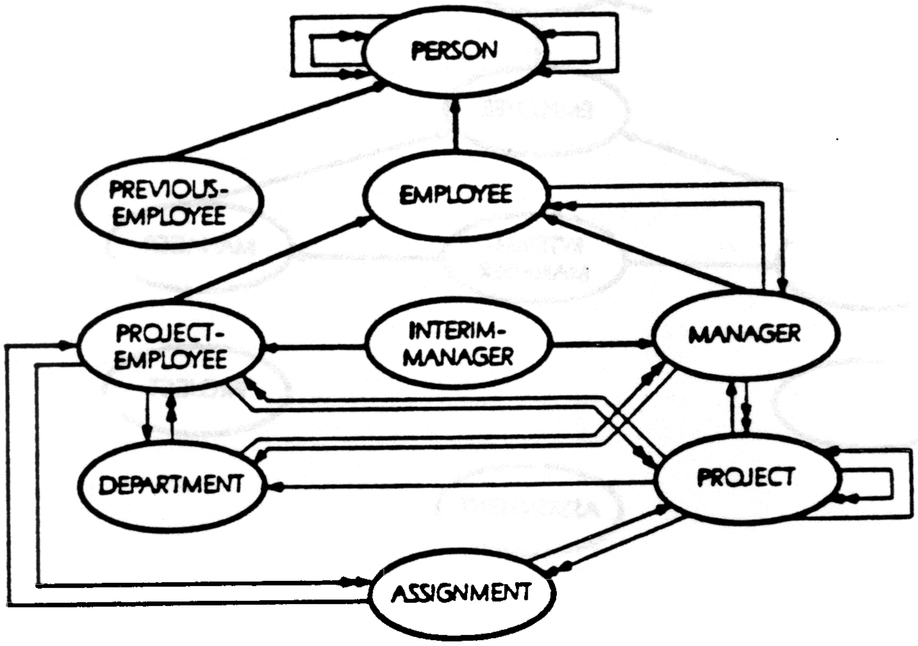
ORGANIZATION - SIM Description



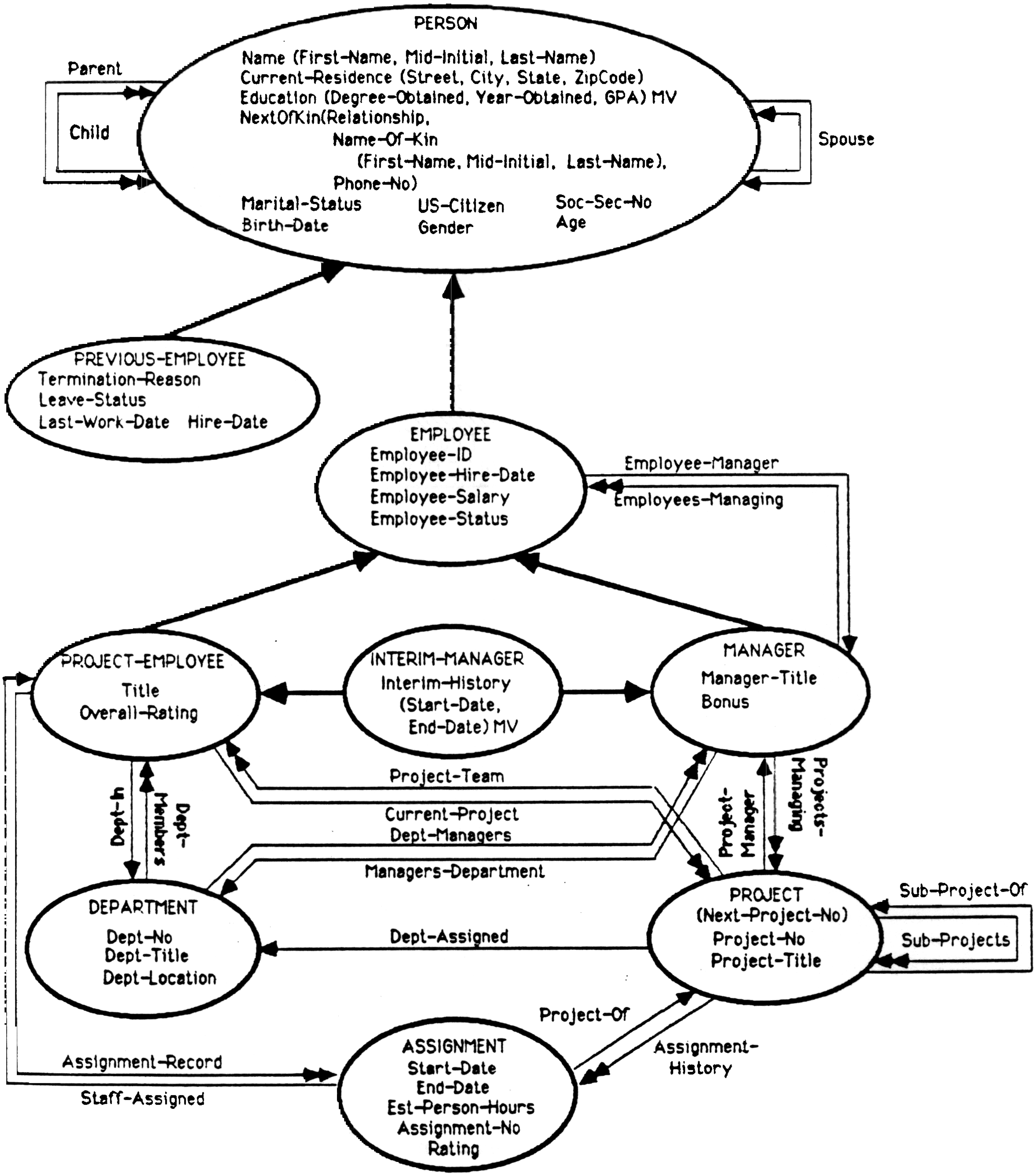
ORGANIZATION - SIM Description

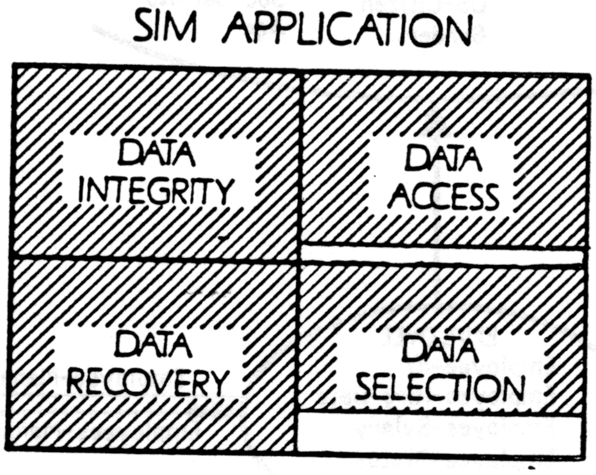


ORGANIZATION - SIM Description



Organization
SIM Description




SIM APPLICATION
DATA INTEGRITY
DATA ACCESS
DATA RECOVERY
DATA SELECTION

ORGANIZATION Data Integrity Enforcement

	Data Item Validation		Referential Integrity	
System	DBMS	Application	DBMS	Application
DMSII	77	25	4	86
	66	36	Ø	9Ø
	1Ø2	Ø	9Ø	Ø

ORGANIZATION COBOL Source Lines

Operation	DMS II	SIM	SIM / DMS II
Population DB	482	227	Ø.47
Retrieve 1	49	44	Ø.9Ø
Retrieve 2	79	42	Ø.53
Modify 1	43	12	Ø.28
Modify 2	76	12	Ø.16
Delete 1	122	12	Ø.1Ø
Delete 2	9Ø	8	Ø.Ø9

Query Examples

Simple Retrieval

Print the names of all non-managers and the title of the department in which they work

Complex Retrieval

Print the names of employees and the titles of all their projects if they work on any project assigned to the Accounting Department

More Complex Retrieval

For the Annual Report Preparation project, print the titles of its subprojects and the names of employees currently assigned

Simple Query

Print the names of all non-managers and the title of the department in which they work

DMSII

```
RELATE Person TO Department BY MATCHING Dept-No WITH Dept-No
AS Person-Dept;

TAB First-Name, Mid-Initial, Last-Name, Dept-Title
WHERE Employed NEQ "Manager"
FROM Person TO Department;
```

Simple Query

Print the names of all non-managers and the title of the department in which they work

DB2

```
SELECT  First-Name, Mid-Initial, Last-Name, Dept-Title
  FROM  Person, Department
 WHERE  Person.Dept-No = Department.Dept-No AND
        Person.Employed NEQ "Manager";
```

Simple Query

Print the names of all non-managers and the title of the department in which they work

SIM

```
RETRIEVE Name of Project-Employee,
         Dept-Title of Dept-In
```

Complex Query

Print the names of employees and the titles of all their projects if they work on any project assigned to the Accounting Department

DMSII

```
RELATE Deptment TO Project BY MATCHING Dept-No WITH Dept-No
AS Dept-Proj;

RELATE Project TO Project-Person BY MATCHING Project-No WITH
Project-No AS Proj-Person;

RELATE Project-Person TO Person BY MATCHING Soc-Sec-No WITH
Soc-Sec-No AS Proj-Emp;

EXTRACT Soc-Sec-No, COUNT AS SSN-Cnt WHERE Dept-Title =
"Accounting" FROM Person TO Project-Person TO Project TO
Department : EXTRACTFILE = Extfile;

OPEN FILE Extfile;

RELATE Extfile TO Project-Person BY MATCHING Soc-Sec-No WITH
Soc-Sec-No AS Extrel;

TAB First-Name, Mid-Initial, Last-Name, Project-Title
FROM Extfile TO Project-Person TO Project
WHERE SSN-Cnt > Ø;
```

Complex Query

Print the names of employees and the titles of all their projects if they work on any project assigned to the Accounting Department

DB2

```
SELECT First-Name, Mid-Initial, Last-Name, Project-Title
  FROM Person, Project-Person, Project
 WHERE Person.Soc-Sec-No = Project-Person.Soc-Sec-No
   AND Project.Project-No = Project-Person.Project-No
   AND EXISTS
     ( SELECT *
         FROM Project-Person, Project, Department
        WHERE Project-Person.Soc-Sec-No = Person.Soc-Sec-No
          AND Project-Person.Project-No =
                 Project.Project-No
          AND Department.Dept-No = Project.Dept-No
          AND Department.Dept-Title = "Accounting")
```

Complex Query

Print the names of employees and the titles of all their projects if they work on any project assigned to the Accounting Department

SIM

```
RETRIEVE Name of Project-Employee,
         Project-Title of Current-Project
WHERE Dept-Title of SOME (Dept-Assigned of Current-Project)
      = "Accounting"
```

More Complex Query

For the Annual Report Preparation project, print the titles of its subprojects and the names of employees currently assigned

DMSII

```
OPEN DMSII Organization;

OPEN DMSII OrgCopy (DMI = DMINTERPRETER/ORANIZATION);

RELATE Project of Organization TO Project of OrgCopy BY
MATCHING Project-No WITH SuperProject-No AS SubProj;

RELATE Project of Organizatoin TO Person of OrgCopy
MATCHING Project-No WITH Project-No AS Project-Emp;

RELATE Project-Person of OrgCopy TO Person of OrgCopy BY
MATCHING Employee-ID WITH Employee-ID AS Proj-Person;

TAB Project-Title of Project of OrgCopy, First-Name of
     Person of OrgCopy, Mid-Initial of Person of
     OrgCopy, Last-Name of Person of OrgCopy
FROM Project of Organization TO Project of OrgCopy TO
     Project-Person of OrgCopy TO Person of OrgCopy
WHERE Project-Title of Project of Organization =
     "Annual Report Preparation";
```

More Complex Query

For the Annual Report Preparation project, print the titles of its subprojects and the names of employees currently assigned

DB2

```
SELECT Project-Title, First-Name, Mid-Initial, Last-Name
  FROM Project, Project SubProj, Person, Project-Person
 WHERE Project.Project-No = SubProj.SuperProject-No
   AND SubProj.Project-No =
          Project-Person.Project_No
   AND Project-Person.Soc-Sec-No = Person.Soc-Sec-No
   AND Project.Project-Title = "Annual Report Preparation"
```

More Complex Query

For the Annual Report Preparation project, print the titles of its subprojects and the names of employees currently assigned

SIM

```
RETRIEVE Project-Title of Sub-Projects of Project,
         Name of Project-Team of Sub-Projects
WHERE Project-Title of Project = "Annual Report Preparation"
```

Host Language Interface

High-level interface to full SIM functionality

Extended language grammar

Transaction orientation

Full data independence

Structured, Tabular, Hybrid retrievals

Language statements may be interleaved with SIM constructs

Language variables allowed in queries

Hybrid Retrieval, COBOL

```
QD DEPARTMENT-QUERY.
Ø1 DEPARTMENT-RECORD.
    Ø2 DEPT-TITLE PIC X(2Ø).

QD MANAGER-QUERY.
Ø1 MANAGER-RECORD.
    Ø2 LAST-NAME PIC X(2Ø).
    Ø2 PROJECT-TITLE PIC X(3Ø).
```

Hybrid Retrieval, COBOL

```
SELECT DEPARTMENT-QUERY FROM DEPARTMENT
     (DEPT-TITLE,
      SELECT MANAGER-QUERY FROM DEPT-MANAGERS
          (LAST-NAME = LAST-NAME OF NAME,
           PROJECT-TITLE = PROJECT-TITLE
               OF PROJECTS-MANAGING)).

RETRIEVE DEPARTMENT-QUERY.

     RETRIEVE MANAGER-QUERY.
```

Hybrid Retrieval, Sample Data

Departments / Managers	Projects
Accounting	
Adams	Payroll
Adams	Year-end Statments
Burns	Accounts Payable
Engineering	
Duncan	
Eaton	Maintenance
Eaton	New Products
Purchasing	
Carr	Inventory

Update: Insert

Create a new employee named John Doe and assign him to the manager named Smith

```
INSERT Employee
     (Name := (First-Name := "John", Last-Name := "Doe"),
      Gender := Male,
      Soc-Sec-No := 123-45-6789,
      US-Citizen := True,
      Employee-ID := 46Ø6Ø,
      Spouse := Person WITH
                (First-Name of Name = "Mary" AND
                 Last-Name of Name = "Doe"),
      Child := INCLUDE Person WITH
               (First-Name of Name = "Junior" AND
                Last-Name of Name = "Doe"),
      Employee-Manager := Manager WITH
                          (Last-Name of Name = "Smith")
     )
```

Promote John Doe to a department manager with a bonus of $5ØØØ

```
INSERT Manager FROM Employee
               WHERE Last-Name of Name = "Doe"
     (Manager-Title := Department-Manager,
      Bonus := 5ØØØ
     )
```

Update: Modify

Reassign all journeyman Project-Employees in the Construction Department to the Maintenance Department

```
MODIFY Project-Employee
     (Dept-In := Department WITH
                 (Dept-Title = "Maintenance"))
WHERE Title = Journeyman AND
      Dept-Title of Dept-In = "Construction"
```

Update: Delete

Remove all projects assigned to managers that manage departments located in Los Angeles

```
DELETE Project
WHERE Dept-Location of Managers-Department
      of Project-Manager = "Los Angeles"
```

Expressions

Allowed in target list and where expression

Automatic null handling (tri-state logic)

Operators

- Arithmetic (+, -, *, /, DIV, MOD, **)
- Boolean (NOT, AND, OR)
- Relational (<, >, =, <=, >=, <>)
- String (&)
- Existence (EXISTS)

Functions

- Arithmetic (ABS, ROUND, TRUNC, SQRT)
- String (LENGTH, EXT, POS, RPT)
- Symbolic (PRED, SUCC)
- Date (YEAR, MONTH, DAY, ELAPSED_DAYS, ADD_DAYS, DAY_OF_WEEK, MONTH_NAME, CURRENT_DATE)
- Time (HOUR, MINUTE, SECOND, ELAPSED_TIME, ADD_TIME, CURRENT_TIME)
- INVERSE
- TRANSITIVE

Pattern matching

Multi-valued Expressions

Operators

- INCLUDE
- EXCLUDE

Functions

- Aggregate (AVG, SUM, COUNT, MIN, MAX)
- Quantifiers (SOME, ALL, NO)

Advanced Query Topics

Subrole attribute

Read-only enumeration of subclasses of a class

May be used in queries

Retrieve the names of employees that are managers

```
RETRIEVE Name of Employee
WHERE Profession = Manager
```

Role testing

Retrieve the names of employees whose spouses are managers

```
RETRIEVE Name of Employee
WHERE Spouse ISA Manager
```

Role qualification

Retrieve the names of US citizens and the employee ID of their spouses

```
RETRIEVE Name of Person, Employee-ID of Spouse AS Employee
WHERE US-Citizen
```

Advanced Query Topics

Multiple perspective queries: Value-based joins

Retrieve the names of project employees and managers that are the same age

```
RETRIEVE Name of Project-Employee,
         Name of Manager
WHERE Age of Project-Employee = Age of Manager
```

Reference variables

Created implicitly by SIM for aggregate functions and quantifiers

Created explicitly in query

Retrieve the names of all managers who manage employees making more than $40,000 and employees making less than $20,000

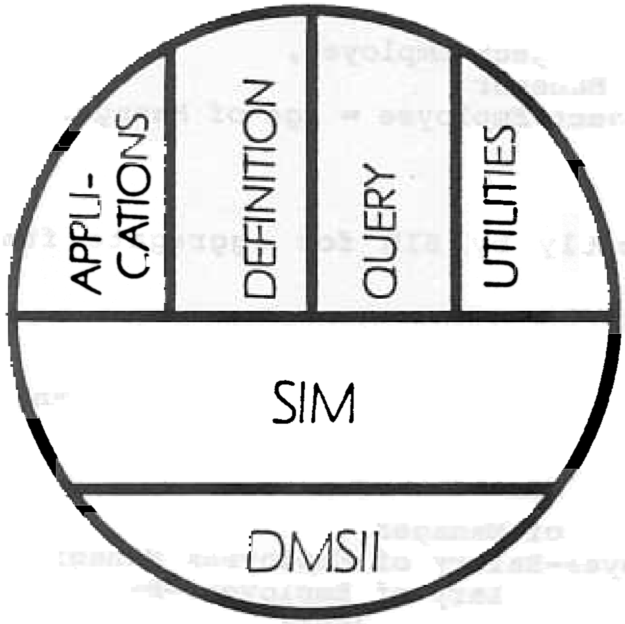
```
RETRIEVE Name of Manager
WHERE Employee-Salary of Employees-Managing > 40000
  AND Employee-Salary of Employees-Managing
      CALLED OtherEmp < 20000
```

Local Selection

Retrieve the names of managers of all departments and the salaries of only the division managers

```
RETRIEVE Name of Dept-Managers of Department
         Employee-Salary of Dept-Managers
             WITH (Manager-Tile of Dept-Managers
                   Division-Manager)
```

InfoExec



PRODUCT OVERVIEW

InfoExec Environment

General

Single, screen-based environment

Seamless, function-oriented screen flow

Screen flows can be record and played back later

Multi-lingual support

All new documentation -- user oriented

Classes available from Joseph & Cogan Associates

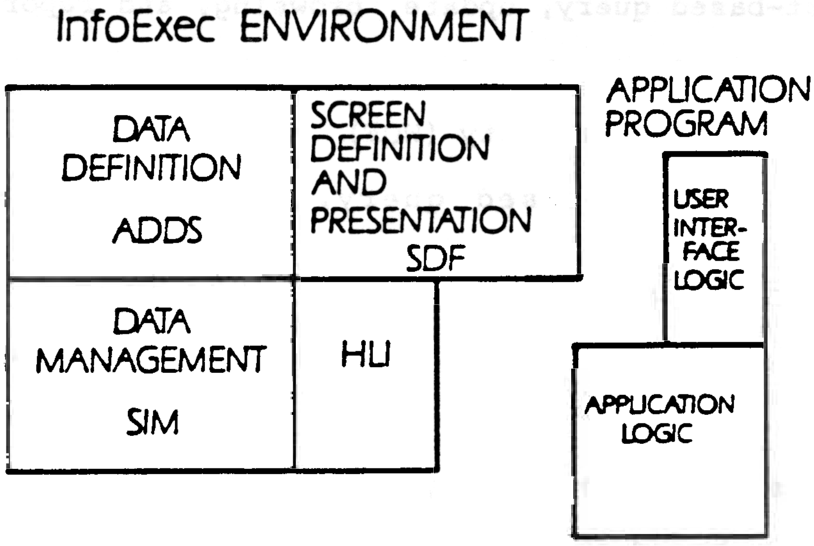
InfoExec Environment

Applications

- Host Language Interfaces
  - COBOL74, Pascal, ALGOL

Definition

- Advanced Data Dictionary System (ADDS)
  - SIM, DMS II, COBOL74, Screen Formats, Saved Queries
  - Logical entities: Program, Process, Keyword
  - Usage tracking and reporting
  - A SIM database application
- Screen Design Facility (SDF)
  - Screen definition and manipulation outside user applicaiton program


InfoExec ENVIRONMENT
DATA DEFINITION
ADDS
SCREEN DEFINITION AND PRESENTATION
SDF
DATA MANAGEMENT
SIM
HLI
APPLICATION PROGRAM
USER INTER-FACE LOGIC
APPLICATION LOGIC

InfoExec Environment

Query

- Interactive Query Facility (IQF)
  - Host-based query, update, browsing, and reporting
  - Screen and menu based
- Workstation Query Facility (WQF)
  - Workstation-based query, update, browsing, and reporting
  - Mouse and window based
  - Unisys B25 and PC families, IBM PC compatibles

Utilities

- Operations Control Manager (OCM)
  - Screen-based operations for SIM and DMS II
- Dictionary Utilities
  - Database schema management and generation
  - Dictionary management and security
  - COBOL74 program loader
- DMS.View and LINC.View
  - Inquiry-only SIM access to existing DMS II databases and LINC systems

SIM Architecture

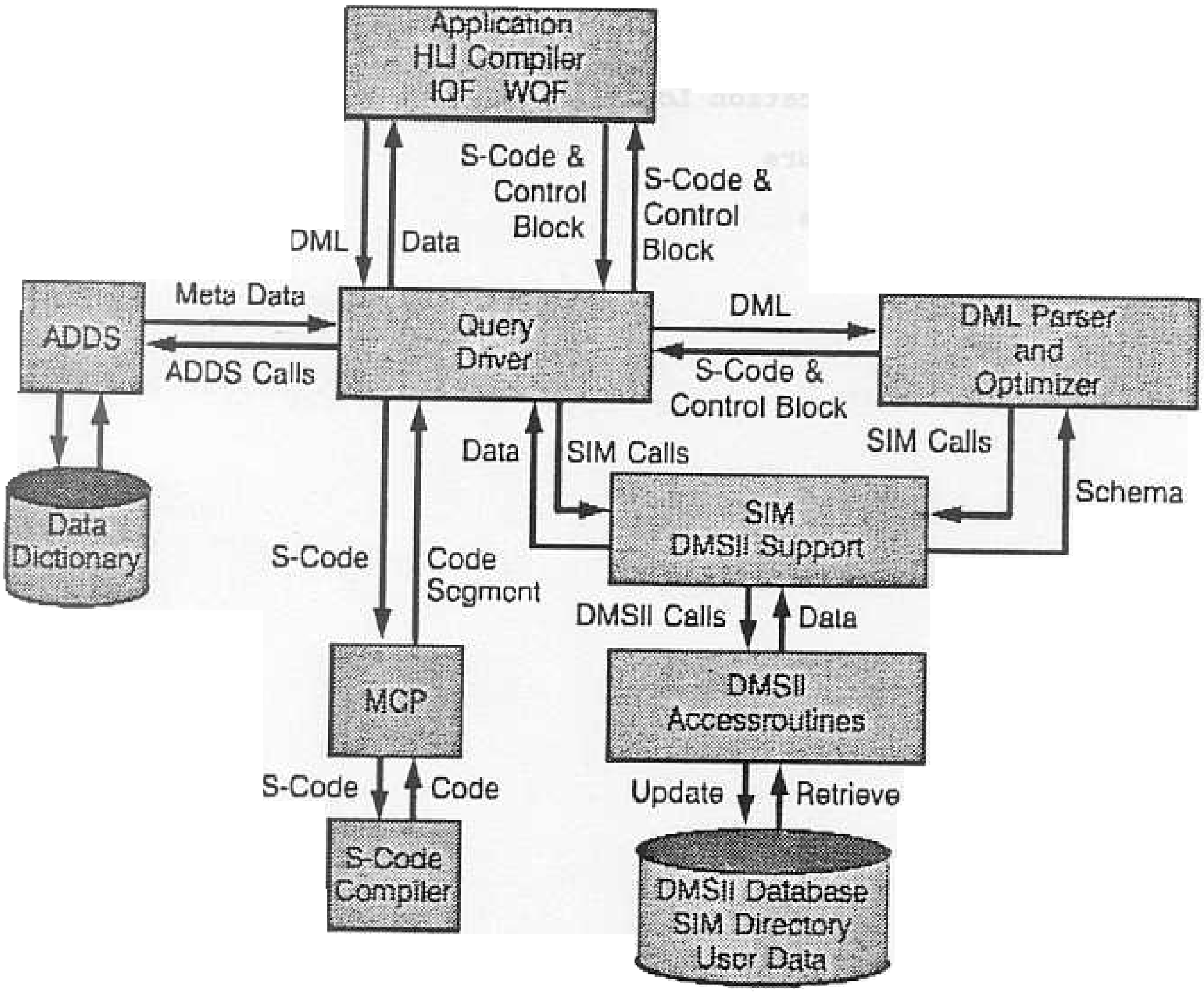
Goals

- Performance
- Date Independence
- Simplified Application Logic

Compilation Architecture

Execution Architecture

SIM Architecture



Interesting Implementation Problems

Bootstrapping

ADDS required to describe SIM schemas, but ADDS is itself a SIM database

DMS II-mapped SIM databases contain an internal, self-describing SIM database

Solution: Special bootstrapping programs

Automatic Reparsing

Some queries may need reparsing and reoptimization between compilation and execution because of conceptual, interface, or physical schema changes

Solution: Queries are timestamped and recompiled as needed

Postponed Updates

Some integrity constraints are enforced during query parsing while others are delayed until execution. Some of the latter require postponing updates until integrity checking is complete.

Solution: Implement update tanking scheme

Interesting Implementation Problems

Locking Protocols

Many interesting locking problems surfaced due to predefined relationships and full referential integrity.

Solution: Implement two-phase locking protocol employing shared and exclusive locks in DMS II.

Performance Tuning

Many mapping options available for each conceptual component, but choice of defaults that perform well in a wide range of applications was not obvious.

Solution: Performance of alternatives studied for surrogates, relationships, and generalization hierarchies.

Host Language Interface

Stylistically difficult to blend query language with host language without destroying data independence.

Solution: Hybrid retrieval.

On-going Issues

Views

- Challenging for strongly typed data models
- May not be as important as in relational model

Derived attributes

System-maintained ordering of classes and EVAs

Dependent classes

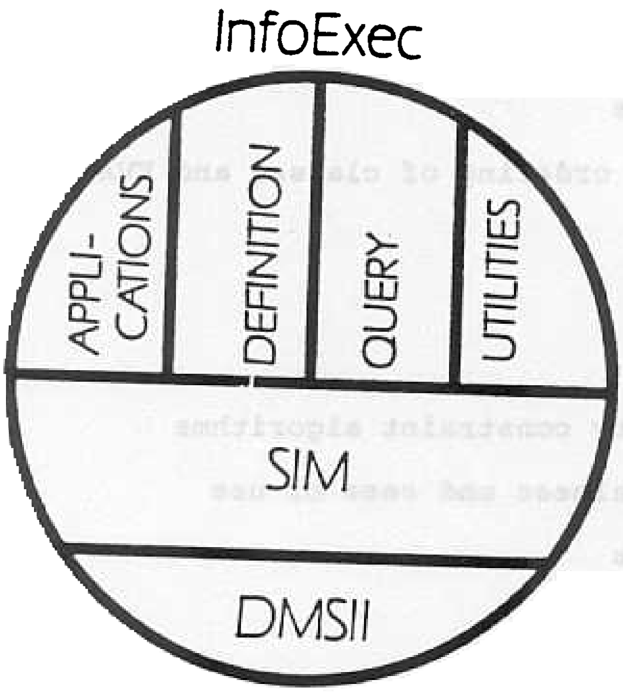
Nested queries

Temporal data

Efficient integrity constraint algorithms

Quantifying naturalness and ease of use

Graphic interfaces



Three Layered Architecture