Active Conceptual Modeling:
New Frontier for Research & Applications

Peter P. Chen

Foster Distinguished Chair Professor

Computer Science Dept., LSU

Baton Rouge, LA 70803, USA

pchen@lsu.edu,

http://www.csc.lsu.edu/~chen

Outline of the Seminar

- Why Conceptual Modeling?
- Review of Basic Concepts in Entity-Relationship (ER) Model
- Needs for Active Conceptual Model (A-CM)
- Mathematical Formulations of a preliminary A-CM
- Another Building Block: Executable Conceptual Models such as
  Microsoft‘s ADO.NET Entity Framework
- Conclusions

Outline (Starting with the 1st Topic)

- Why Conceptual Modeling?
- Review of Basic Concepts in Entity-Relationship (ER) Model
- Needs for Active Conceptual Model (A-CM)
- Mathematical Formulations of a preliminary A-CM
- Another Building Block: Executable Conceptual Models such as

  Microsoft's ADO.NET Entity Framework
- Conclusions

Why do we build conceptual Models?

- In problem solving, a well known advice/lesson is:

"If you can understand, state, and organize the problem clearly, you are already half-way in solving the problem."

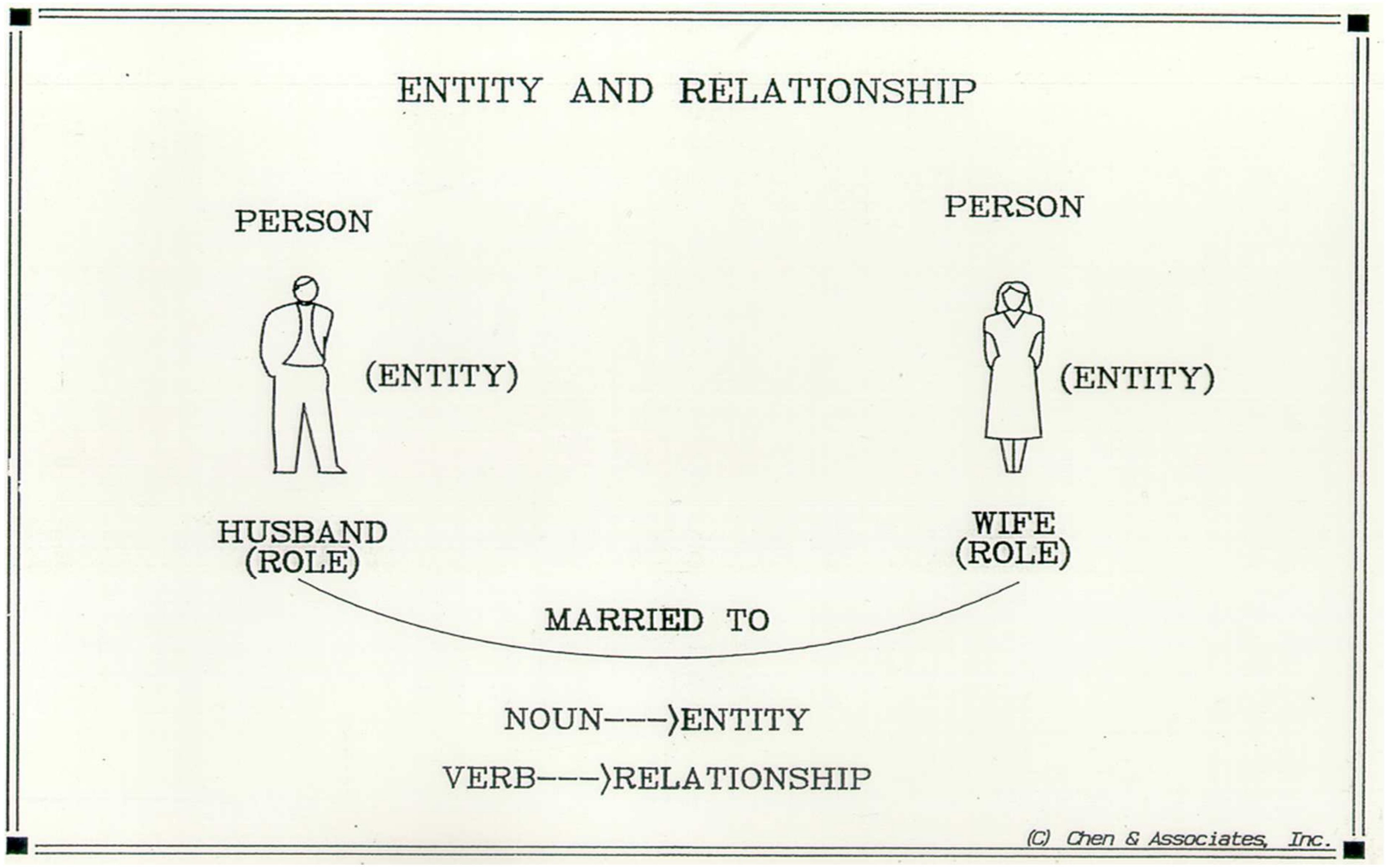
- A clear and organized conceptual model reflects your understanding of the problem at hand.

Outline

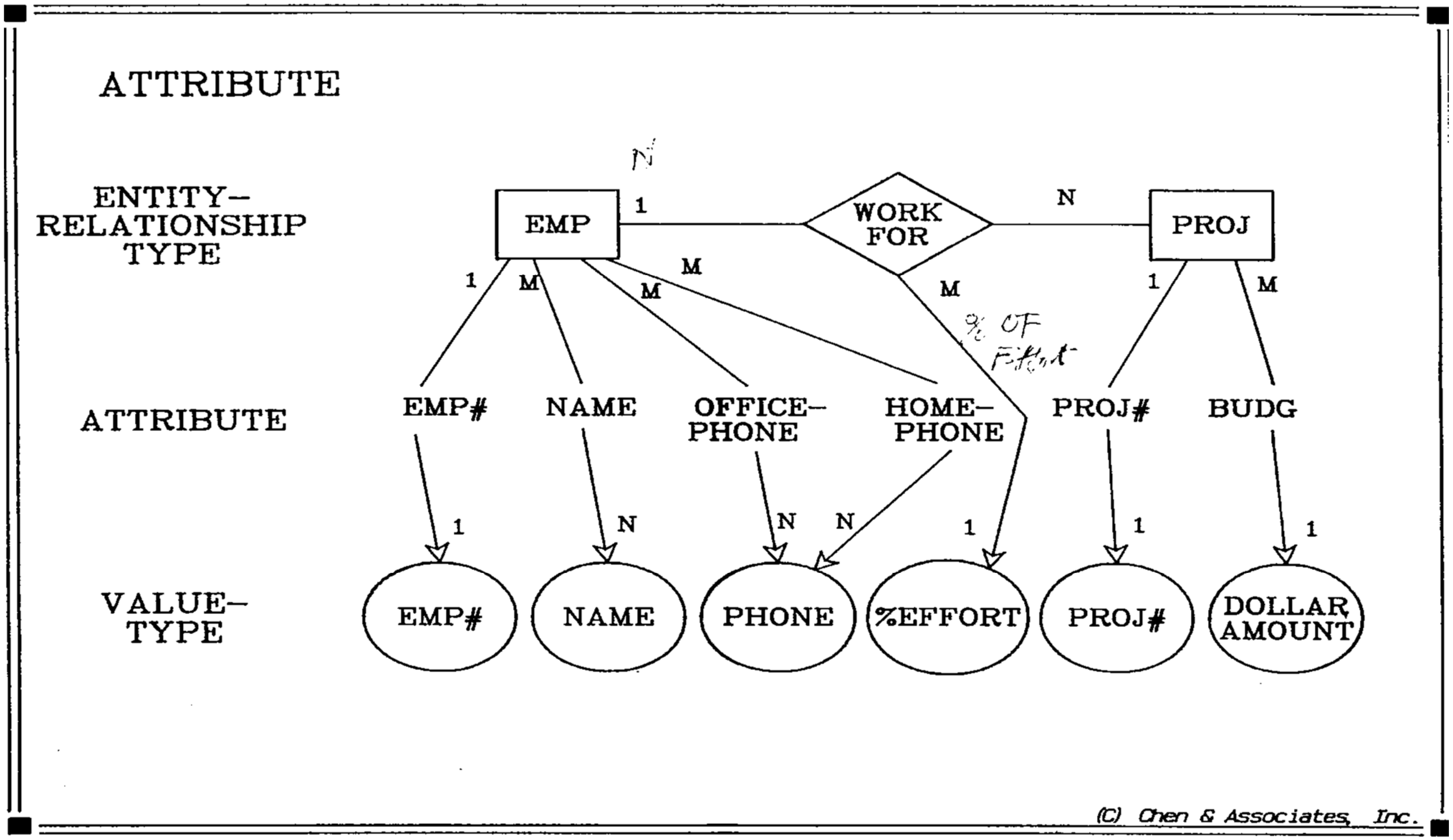
- Why Conceptual Modeling?
- Review of Basic Concepts in Entity-Relationship (ER) Model
- Needs for Active Conceptual Model (A-CM)
- Mathematical Formulations of a preliminary A-CM
- Another Building Block: Executable Conceptual Models such as

  Microsoft's ADO.NET Entity Framework
- Conclusions

Concepts of Entity & Relationship



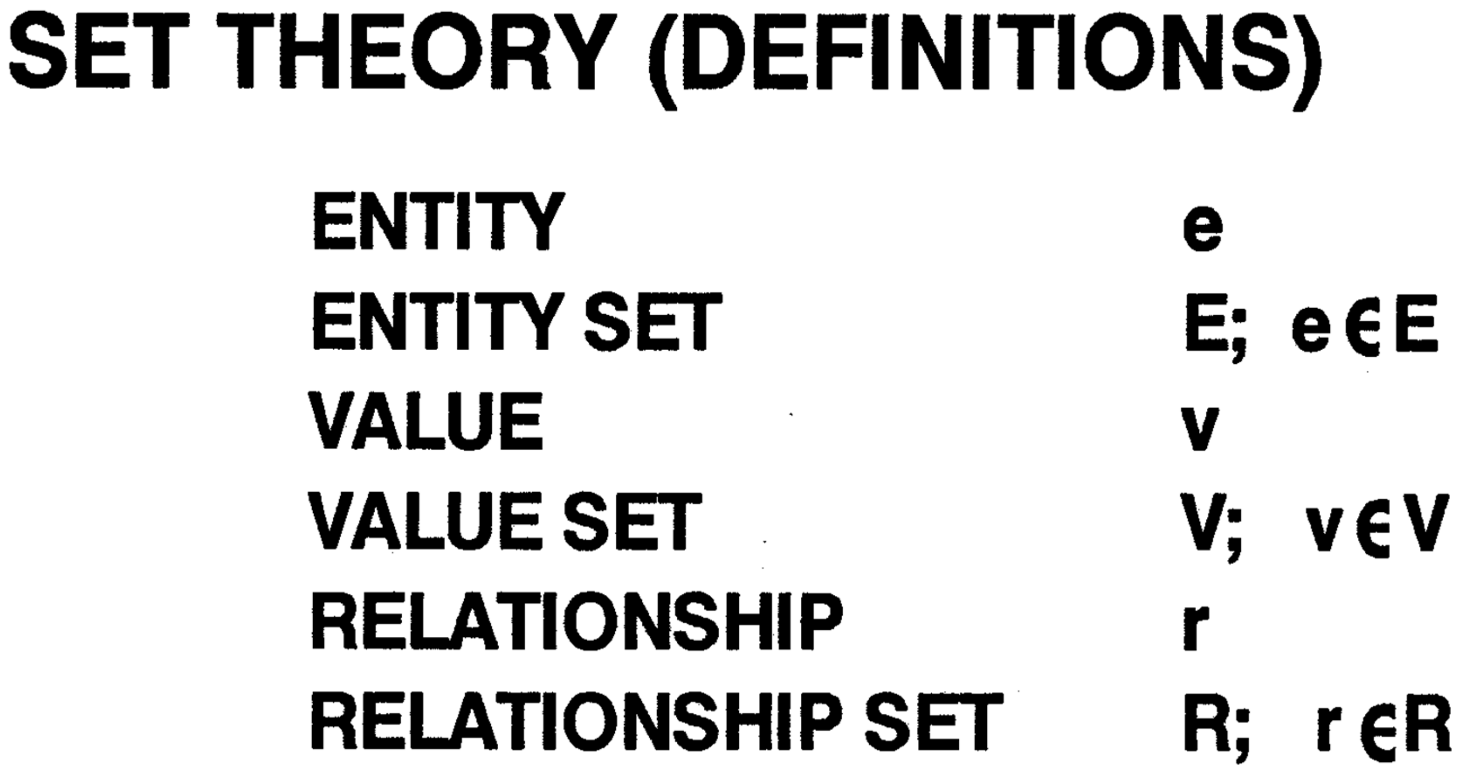
An Example of ER Diagram



Theoretical Foundations of ER Model

- Set Theory
- Modern Algebra
- Logic
- Lattice Theory

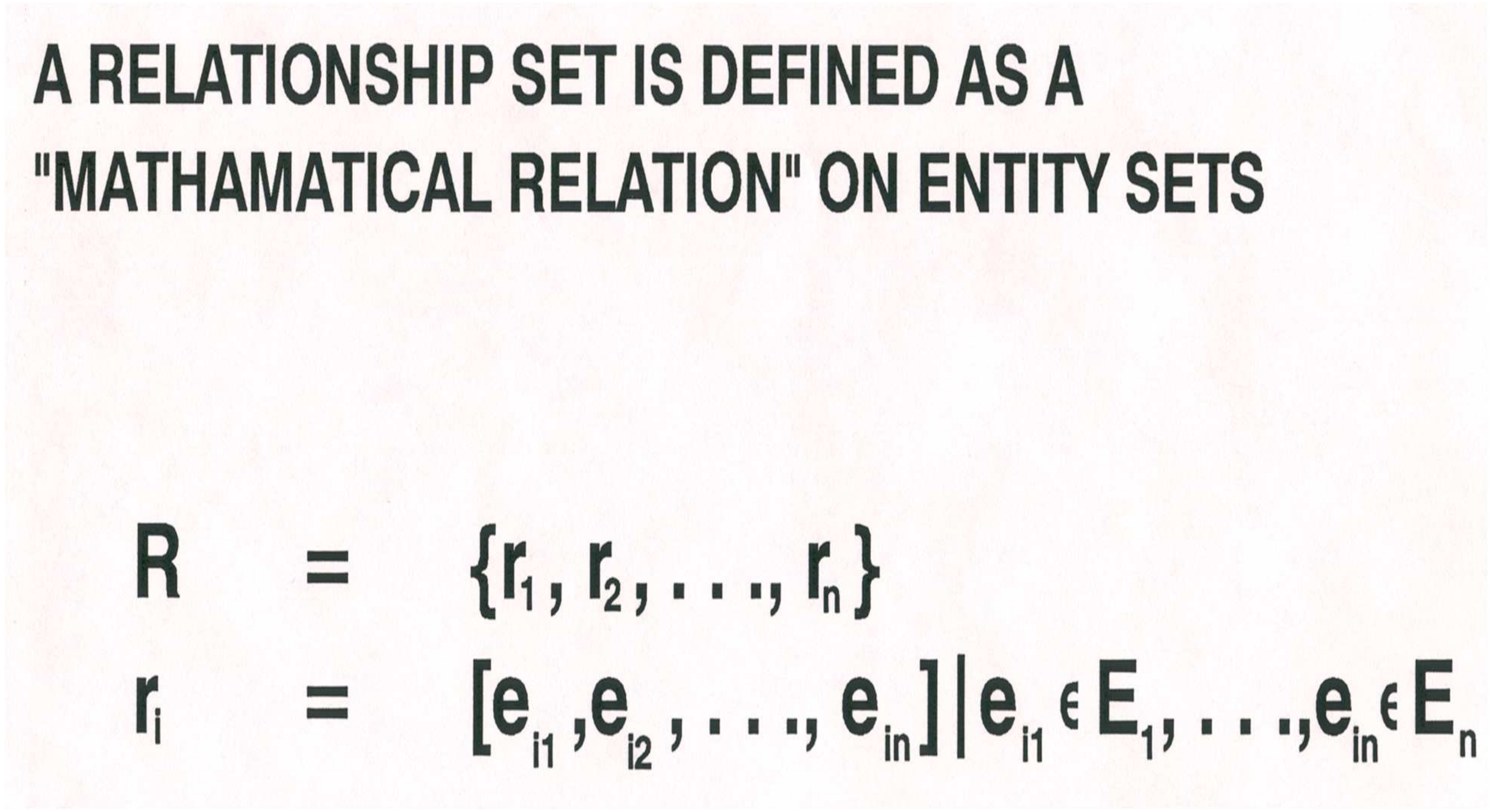
Defining ER Concepts using Set



SET THEORY (DEFINITIONS)

ENTITY	e
ENTITY SET	E; $e \in E$
VALUE	v
VALUE SET	V; $v \in V$
RELATIONSHIP	r
RELATIONSHIP SET	R; $r \in R$

Relationship as an Ordered Tuple



A RELATIONSHIP SET IS DEFINED AS A "MATHAMATICAL RELATION" ON ENTITY SETS

$$R \quad = \quad \{r_1, r_2, \ldots, r_n\}$$

$$r_i \quad = \quad [e_{i1}, e_{i2}, \ldots, e_{in}] \mid e_{i1} \in E_1, \ldots, e_{in} \in E_n$$

The Present Status of Conceptual Modeling (I)

- ER Modeling is the „most widely used methodology“ in the business DB application development world
  - A very large percentage of companies and government agencies are using it
- Most CASE tools support ER Modeling
  - Oracle Desinger – ER Diagrammer
  - Computer Associates ERWIN
  - Sybase Power Designer
  - Microsoft Access, Visio
- UML (Unified Modeling Language) reinforces the ER concepts

The Present Status of Conceptual Modeling (II)

- OO Modeling incorporates many concepts of ERM
  - However, „object“ is an implementation concept
  - Current OO methodologies need more general concepts of „relationship“

- „Data Mining“ is an implicit way of constructing ER models from data
  - Discover hidden „relationships“

    Discover the embedded ER Models

The Present Status of Conceptual Modeling (III)

- The ER model triggered a new field of study -- Conceptual Modeling
- ER Conferences are being held around the world as an Annual Conference on Conceptual Modeling:
  - 2001 in Japan; 2002 in Finland
  - 2003 in Chicago; 2004 in Shanghai, China
  - 2005 in Austria; 2006 in Tucson
  - 2007 in New Zealand; 2008 in Barcelona
  - 2009 in Brazil; 2010 in Vancouver
  - 2011 in Brussels

Outline

- Why Conceptual Modeling?
- Review of Basic Concepts in Entity-Relationship (ER) Model
- Needs for Active Conceptual Model (A-CM)
- Mathematical Formulations of a preliminary A-CM
- Another Building Block: Executable Conceptual Models such as

  Microsoft‘s ADO.NET Entity Framework
- Conclusions

The Needs for Active Conceptual Modeling

A Major Need: An Active Conceptual Model (ACM).
In particular, an "Active Conceptual Model" of Learning" for analyzing surprises, crises, and unconventional events.
Surprising incidents and crises force us to look back to the past changes of our world situation from a global perspective:

- September-11, Hurricane Katrina in the U.S.
- Tsunami, Earthquakes around the world

A framework for documenting and analyzing surprises and crises:

- *How do we analyze the surprise/crisis scenarios?*
- *What information do we need to analyze the surprise/crisis situations?*
- *What have we learned from the surprises/crises?*
- *How can we handle surprises/crises in current and future world situations?*

Problems of Existing Methodologies/Technologies

- Current databases/KBs, which describe our knowledge of a domain as snapshots, usually do not support information and schema changes or historical information
- Current state-of-art techniques focus on pre-defined entities of interest and their static relationships
- Very few constructs are available for modeling changes of the entity behaviors (e.g. terrorist profiles), and the dynamic and time-varying relationships among them
- A wide spectrum of situations resulting from different degrees of importance of the relationships from different perspectives are difficult to be represented

Proposed Solution:
Active Conceptual Modeling (ACM)-1

- A continuous process of describing all aspects of a system or domain, its activities, and its changes under different perspectives, based on our knowledge and understanding

- The active conceptual model will provide the necessary needs of control and traceability for the evolving and changing world state

- Help understand relationships among changes which may have significance to current world state (e.g. Terrorist training methods could have been changed since 9-11)

Proposed Solution:
Active Conceptual Modeling (ACM)-2

- Conventional conceptual modeling can be viewed as a simple case of active conceptual modeling

- Allows for continuous learning and provides traceable lessons learned from past experiences, surprises, and crises.

- Hopefully, it will be potentially useful for predicting future

Some General R&D Issues of A-CM

(1) Extensions of

(a) Concepts,

(b) Diagrammatical Techniques

(c) Mathematical Theory

(*In the next few slides, we will review the current status and discuss what are needed to be developed)*

(2) Software

(a) Design Tools

(b) Application/code generators

(c) Executable conceptual model

(*We will discuss further on "Executable Conceptual Model")*

Extensions of Concepts, Diagrammatic Techniques, and Mathematical Theory (1)

- Extending/Refining the Entity-Relationship (ER) Model to handle the following concepts
  - The “Time” and “Space” Dimension
  - “Scenario” Description
  - “Roles” of the players
  - “Cause/Effect” Relationship
  - “Event/Activity/Time” Symbols (Icons) and Interrelationships
  - Etc.
- Some of these issues have been studied in the past, but we need coherent and integrated solutions!

<u>Extensions of Concepts, Diagrammatic Techniques, and Mathematical Theory (2)</u>

- Extending the ER Model from the User/Operation Perspectives
  - Represent a given snapshot of the real world by a mathematical model
  - Representing the differences between snapshots by a “delta” model
  - Create a database, and the users can query this database to study the status of the world state with respect to the changes and their relationships

Outline

- Why Conceptual Modeling?
- Review of Basic Concepts in Entity-Relationship (ER) Model
- Needs for Active Conceptual Model (A-CM)
- Mathematical Formulations of a preliminary A-CM
- Another Building Block: Executable Conceptual Models such as

  Microsoft‘s ADO.NET Entity Framework
- Conclusions

Motivations and Applications(I)

(A) Need for a more rigorous specification and evaluation technique for systems, particularly for large complex systems

- Not just use qualitative reasons such as:
  - “this system architecture is more flexible”
- Not just use simulation methods

- Hopefully, this more rigorous technique can complement qualitative reasons and simulation methods to make a more complete and comprehensive specification and evaluation methodology

- Example of Military Impact Areas:
  - Large scale system acquisition programs
  - Design of future systems
  - Specifications of maintenance of existing systems

Motivations and Applications (II)

(B) Need for a small, consistent, easy-to-understand integrated modeling methodology which covers static and dynamic situations

- Weaknesses of Current Modeling Methodologies:
  - The Entity-Relationship (ER) Model has been widely in database and software engineering, but it tends to emphasize the static modeling
  - The UML is a collection of many modeling techniques and seems to be too large and too complex for average systems analysts and developers.

Motivations and Applications (II)

- (C) Need for the architect and design of a system to predict the future “surprises” better
  - Problems of existing systems:
    - Many isolated systems which only records the most current facts
    - Very little historical data and documentation of reasons/events for the changes
    - Very difficult to predict the future
    - Very difficult to prevent the happening of the undesired events
    - Take a long time to find out the cause-effects which triggered and derived the “surprises”

Some of the Previous Work

- Existing System Modeling/Simulation Methodologies and languages
- In database and software engineering fields:
  - Entity-Relationship (ER) model [Chen, ACM Trans. On Database Systems, Vol. 1. No.1, 1976]
  - Which triggered:
    - Other semantic data models
    - Computer-Aided Software Engineering (CASE) Methodologies and Tools
    - Object-Oriented Analysis and Design Methodologies, UML
- An Algebra for Binary Directional ER Model [Chen, IEEE 1st Data Engineering Conference,1984], a “time” dimension paper and two “Entity lattice” papers by Chen
- Many extensions and modifications of the ER model by researchers around the world

Algebraic Operators

- A preliminary set of algebraic operators has been defined for specifying and analysis of architectures
  - Defined a set of algebraic operators
  - Interesting (and somewhat innovative) features:
    - Operators have restrictions on conditions of applications
    - Operators incorporate costs, time,
  - Investigation of how to modeling the "Time"
- Investigate the applications of this set of operations in several settings:
  - Human (commercial or military) organization analysis
  - Description and Prediction of Future Surprises
  - Building a Theory of "Social Networks" and "6-degrees" of relationships of people

The core of "Active Conceptual Model": A Proposed Algebra System

- Three Types of Sets
  - Entity Sets, Relationship Sets, and Value Sets
  - Relationship is a "mathematical relation" defined on a Cartesian Products of Entity Sets
  - Attributes are "mappings/math. relations"
- A Set of Algebraic Operators
  - Operations on Entity Sets
  - Operations on Relationship Sets
  - Operations on Value Sets
- Innovative/New Features
  - Operators have "cost functions," "time duration function," "pre-conditions," "after-conditions," etc.

Mathematical Foundations of ER Model

ER model (based on set theory, mathematical relations, Modern algebra, logic, and lattice theory)

- Entity Set/Entity E; $e \in E$
- Relationship Set/Relationship R, $r \in R$ - mathematical relation defined on Cartesian Products of Es

  $R = \{r_1, r_2, \ldots, r_n\}$ $\quad r_1 = \{e_{i1}, e_{i2}, \ldots, e_{in}\} \mid e_{i1} \in E_1, \ldots, e_{in} \in E_n$
- Value set/Value V; $v \in V$
- Attribute: a function which maps from E or R into V $F: E_i$ or $R_i \rightarrow Vi$ or $V_{i1} \times V_{i2} \times \ldots \times V_{in}$
- Algebraic operators:
  - Selection of entities, Union of Es, Difference of Es
  - Selection, union, difference, and intersection of Vs
  - Creation, composition, decomposition, inverse of relationships
  - Cartesian product of Vs
- ER extensions

Examples of Algebraic Operators

- Composition of relationships:
  - Parents (Parents (x: person))
    = grandparent (x: person)
- Construction of a high-level entity (i.e., assembly) from several low-level entities (components):
  - W = Construct ([x, y, z,…], where x, y, z are … and cost functions, conditions, constraints)
- Deletion of Relationship
- Addition of Relationship
- Move an entity = break up relationship(s) and addition of (new) relationship(s)

What types of critical issues can this proposed algebra system help?

(1) How can one define "flexibility“, "complexity“ and other “qualitative” measures/factors analytically?

(2) Based on the definitions in (1), how can one find the optimal point to switch from one structure to another?

(3) What are the algebraic operations to convert one structure to another?

(4) Others

Outline

- Why Conceptual Modeling?
- Review of Basic Concepts in Entity-Relationship (ER) Model
- Needs for Active Conceptual Model (A-CM)
- Mathematical Formulations of a preliminary A-CM
- Another Building Block: Executable Conceptual Models such as

  Microsoft's ADO.NET Entity Framework
- Conclusions

Executable Conceptual Model – Historical Perspectives (1)

- In the early days (During late 70’s and early 80’s)
  - ER Languages and Prototypes were proposed and developed
  - A database company called INGRES had a serious internal discussion on whether to implement an ERDBMS or not
  - Bell Labs implemented and operated an ER DBMS for a phone maintenance system

Executable Conceptual Model – Historical Perspectives (2)

- During late 80’s and early 90’s)
  - Software AG had a DBMS product called ENTIRE (ENTity RElationship).
  - IBM researchers published and implemented ERLANG (ER LANGuage) software prototype
  - ANSI adopted ER model as the meta model for IRDS
  - As a part of the AD-Cycle, IBM offered a product, DB2 Repository, based on the ER model.
  - Digital had CDD-plus, based on the ER model
- The above were not commercially successful
- Some were commercially successful:
  - ZIM (an ERDBMS developed in Canada) was the best selling DBMS in Brazil
  - SAP used the ERM to integrate software modules

Executable Conceptual Model – Current Status

- Now, a new surge of interest – for example,
- **A product from Microsoft!**
  - Jose Blakerley, S. Muralidhar, & A. Nori, "The ADO.NET Entity Framework: Making the Conceptual Level Real", Proc. ER Conference 2006.
  - A Adya, JA Blakeley, S Melnik, "Anatomy of the ado. net entity framework, Proc. of SIGMOD, 2007**.**
  - JA Blakeley, V Rao, I Kunen, A Prout, ".NET database programmability and extensibility in Microsoft SQL server," Proc. SIGMOD, 2008

Executable Conceptual Model – Current Status (Continued)

- Jose Blakerley, S. Muralidhar, & A. Nori, “The ADO.NET Entity Framework: Making the Conceptual Level Real”
  - **Supports SQL-Server**
  - **4- level architecture:**
    - **Presentation/programming level: XML, Object, etc.**
    - **Conceptual level: Entity**
    - **Logical level: Relational**
    - **Physical level**

ADO.NET Entity Framework

Note: The following few slides on Entity Framework came from:

José Blakeley, S Muralidhar, Anil Nori

SQL Server

Microsoft Corporation

Overview

Impedance mismatch	• Applications and data services
Entity Framework	• EDM and Entity SQL • Making the conceptual level real
ADO.NET V3	• Mapping engine • Object Services
Summary	

Problem: Impedance Mismatch

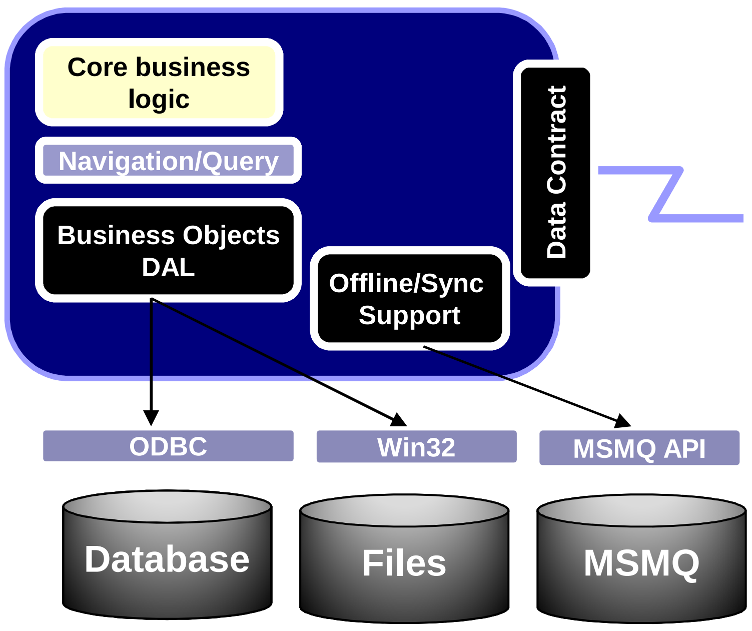
- For applications

  Objects ≠ Rows

- For data services

  Entities ≠ Rows

Impedance Mismatch in Apps



Deal in terms of "entities"

- Data flowing between composite applications is not rows

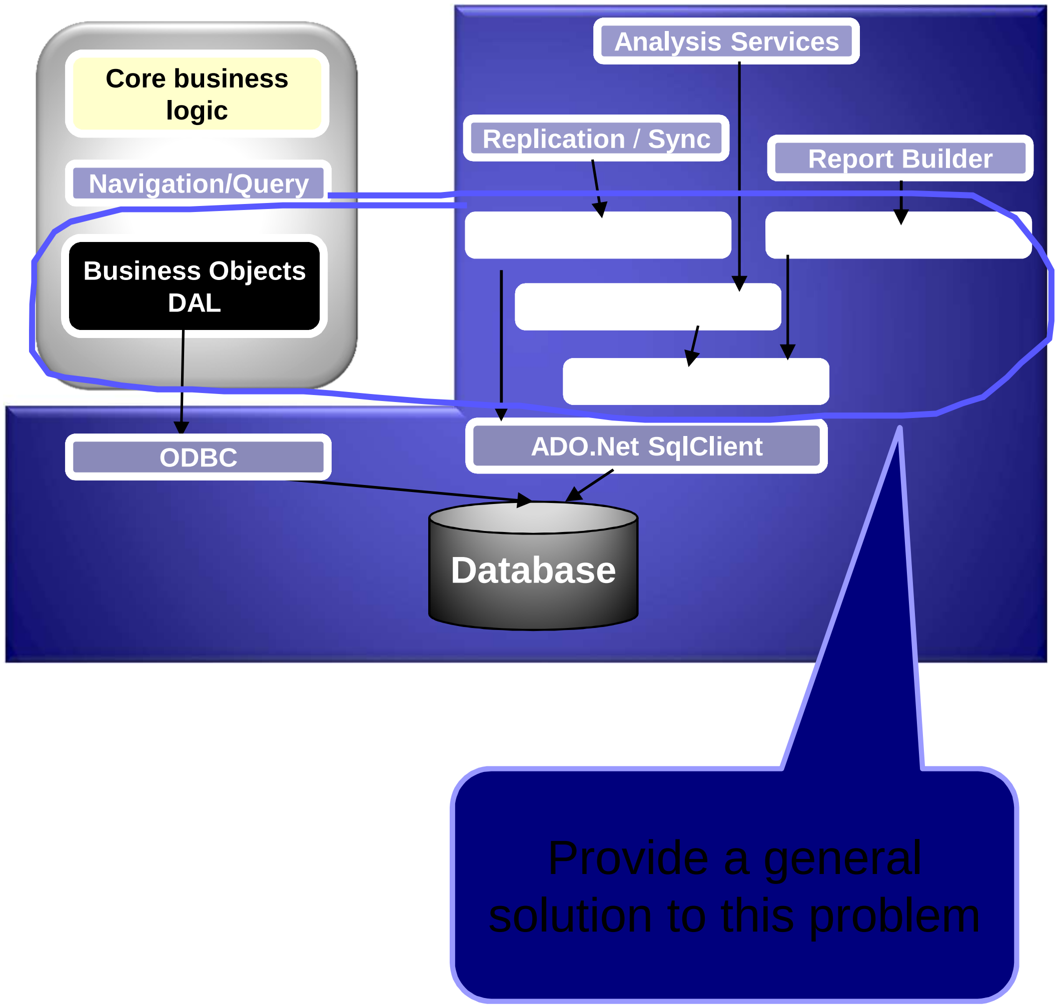
Perform many data transformations

- Persistence, workflow, serialization

Overcome impedance mismatch on their own

- Apps bundle their own navigation & query transformation
- Up to 40% app code is data access

Impedance Mismatch in Data Services



Provide a general solution to this problem

Applications build “entities” in their DAL layer

- Customers, accounts, etc.

DBMS Data Services work on entities too

- Replication, reporting, business intelligence have their own way to describe them

Can’t directly get value from data services without redefining entity each time.

Solution

Define high-level conceptual model

- Entity data model → entities & relationships
- Separate value and object layers

Define query language for conceptual model

- Entity SQL

Build a mid-tier view manager – makes the conceptual model concrete

- Powerful E-R mapping – bidirectional views
- Query and update processing
- Apps and services can program to the conceptual layer
- Object services are layered above the conceptual (entity) abstraction → ORM functionality

Language integrated query

- Compile time type checking
- Integration with development environment - IntelliSense

Entity Data Model

- Borrows from Relational and E-R models
- Types
  - Entity Type, Relationship Type
  - Complex types, Inheritance
- Instances
  - Entity Set, Relationship Set
  - Entity Containers - database instance
- Entity SQL to query against EDM sets

EDM schema language

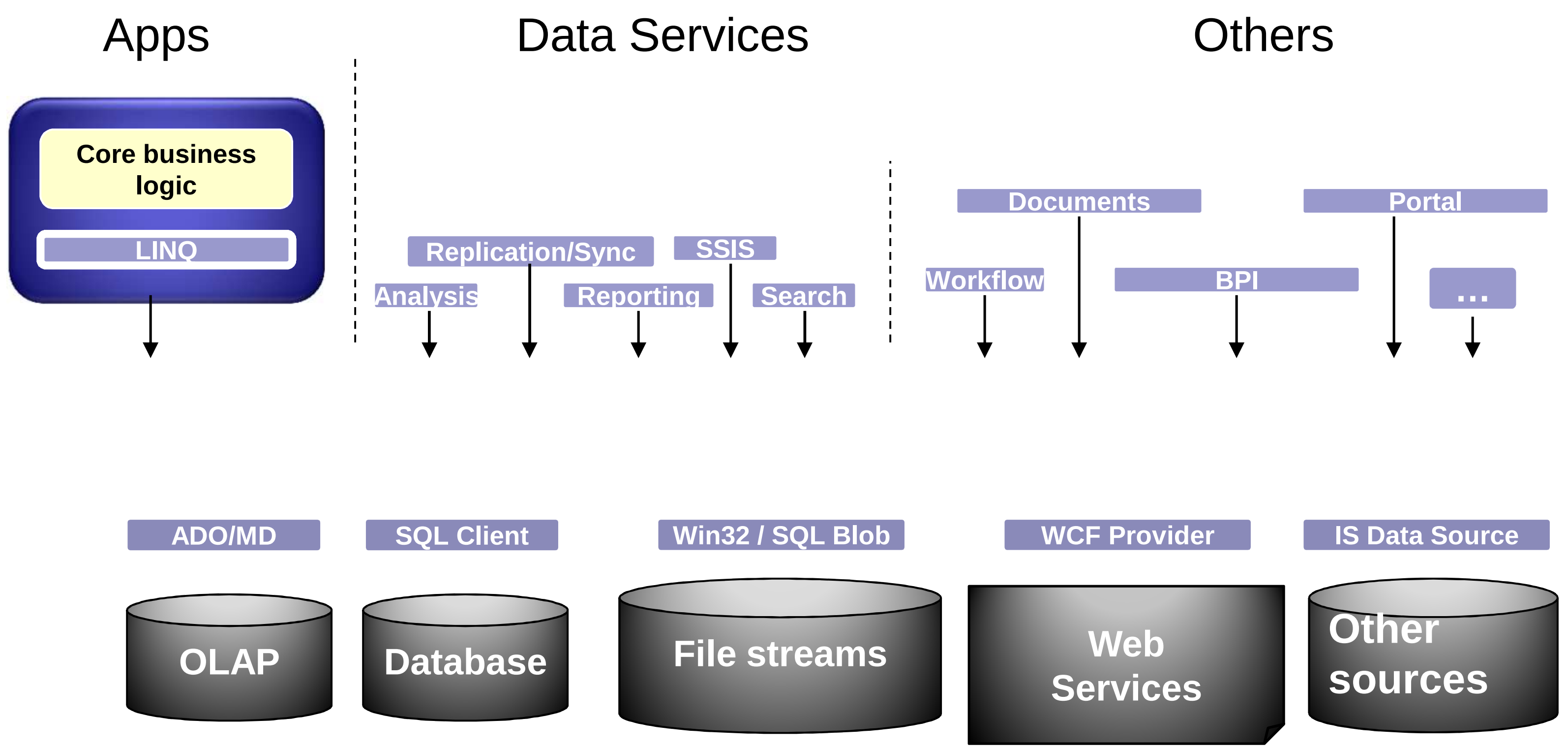
- **Types:** Define the shape of instances
  - Scalar – e.g., integer, datetime, string
  - Complex types – e.g., Address
  - Row types (created to represent eSQL results)
  - **Entity types:** Include identity
    - Inheritance
  - **Relationship types:** Describe associations among entities
    - Multiplicity and operational constraints
- **Instances**
  - **Entity sets:** Container for entities of particular entity type
    - May be mapped to one or more store tables
  - **Relationship sets:** Container for pairs of entity references
    - May be mapped to foreign key fields or join store tables
- EDM schemas are independent of store schemas and programming language type systems

EDM schema language

("fit-on-a-slide" syntax)

```
complexType ContactInfo{ emailaddress: String; phone: String};
entity type SalesOrder
  type { id : Int32;
         orderDate : DateTime;
         status”: Byte;
         accountNumber : String;
         totalDue”: Decimal }
  key { id };
entity type StoreSalesOrder : SalesOrder
  type { tax: Decimal }
entity type SalesPerson
  type { id : Int32; ...
         contacInformation: ContactInfo }
  key { id };
relationship type SalesPerson_Order{
  salesPerson: SalesPerson[1];
  order: SalesOrder[0+]
};
entity set SalesPeople over SalesPerson;
entity set SalesOrders over SalesOrder;
relationship set salesPersonOrders over SalesPerson_Order;
```

ADO.NET Entity Framework



EDM raises the level of abstraction around data.

EDM models can be augmented with behavior, UI, and other aspects

EDM describes the nouns that other platform services operate on

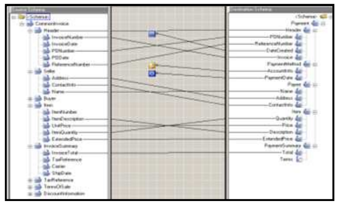
EDM allows us to move key services from application logic to the platform.

EntityConnection

EntityCommand

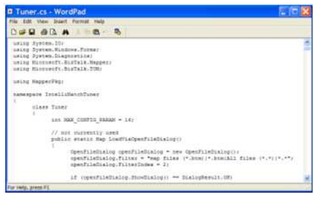
EntityDataReader

Powerful E-R Mapping


Correspondences
(specified by users)

```
Select ord#, prod#,
cust#
From  Shipped
⊆
Select ord#, prod#,
cust#
From Order Join Item
on ord#
```

Mapping



```
using System.IO;
using System.Windows.Forms;
using System.Diagnostics;
using Microsoft.BizTalk.Mapper;
using Microsoft.BizTalk.TOM;

using MapperPkg;

namespace IntelliMatchTuner
{
	class Tuner
	{
		int MAX_CONFIG_PARAM = 16;

		// not currently used
		public static Map LoadViaOpenFileDialog()
		{
			OpenFileDialog openFileDialog = new OpenFileDialog();
			openFileDialog.Filter = "map files (*.btm)|*.btm|All files (*.*)|*.*";
			openFileDialog.FilterIndex = 2;

			if (openFileDialog.ShowDialog() == DialogResult.OK)
```

Query and Update Views
(drive the runtime)

- Declarative mapping definition
  - Allows non-expert users to specify complex mappings
  - Formal semantics (vs. *ad hoc* format)
  - Enables impact analysis and schema evolution

- Bidirectional views
  - Compiled automatically from mapping
  - Uniform, efficient runtime
  - Simplify dev & test (vs. case-by-case)
  - Leverage robust DB technology
  - Solve “view update” problem

Summary

Entity Framework and Language Integrated Query

- Eliminate the impedance mismatch for apps & data services
- Clean separation between rich value and object layers

Entity Data Model

- Raises the level of abstraction around data
- Schemas can be augmented with behavior, constraints, UI, and other aspects
- Describes the nouns that other data services operate on
- Allows moving key data services from application logic to data platform

Entity SQL

- Provides query capability for EDM
- SQL + nest, unnest, inheritance, relationship navigation

ADO.NET

- Runtime for EDM and Entity SQL → makes conceptual layer real
- Sound mapping engine
- Object services - typed queries, materialization, change tracking
- Language integrated query
- Objects when appropriate, explicit control when necessary

Outline

- Why Conceptual Modeling?
- Review of Basic Concepts in Entity-Relationship (ER) Model
- Needs for Active Conceptual Model (A-CM)
- Mathematical Formulations of a preliminary A-CM
- Another Building Block: Executable Conceptual Models such as

  Microsoft‘s ADO.NET Entity Framework
- Conclusions

Conclusions (I)

- ER Modeling was triggered by critical needs
  - Unifying data views from top-down and bottom-up perspectives
  - For vendors & user organizations
  - Incorporating more sematics
- Entity and relationship are fundamental concepts for
  - Data/Knowledge Representation
  - Database design
  - Software engineering
  - Information system development
  - And others (data mining, system modeling/specifications, etc.)

Conclusions (II)

- Active Conceptual Modeling
  - Is the Next Major Development of Conceptual Modeling
  - Can Help Us Understand the Relationships of Past Events and Make Better Decisions for Future Events
  - Extending/Refining the ER Model is a Feasible and viable Solution
  - Executable conceptual model is becoming a reality!

The End

- Thank You!