


Engineering Artifacts for Multi-Agent Systems

(Version 2)



Van Parunak, ERIM CEC (vparunak@erim.org)
Jim Odell, James Odell Associates
(jodell@compuserve.com)




Van Paranak

- **Scientific Fellow, ERIM.**
- **Senior technical direction of marketing initiatives and projects**
- **Developed and executed a strategy for coherent identification, development, and marketing of ERIM tools**
- **Responsible for marketing to the research community.**
- **www.erim.org/~van/**

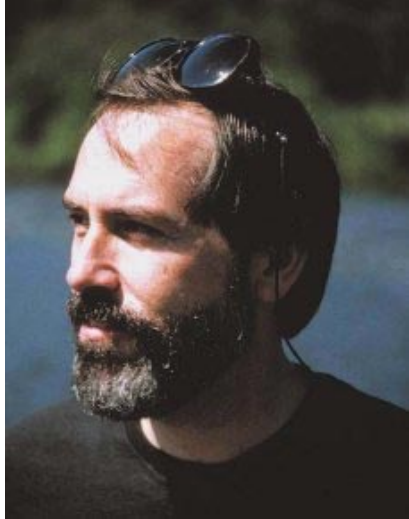


10/10/99 Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved. 2




James Odell

- **Industrial Consultant in OO, Agents, and Complex systems**
- **Based in Ann Arbor, MI**
- **Leadership role in OMG:**
 - **Co-Chair of Object Analysis and Design Task Force (source of UML)**
 - **Co-Chair of Agents Working Group**
- **www.jamesodell.com**



10/10/99Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.3



Overview

- **Why Engineering Artifacts?**
- **OO Artifacts for MAS Development**
- **Toward Agent-Specific Artifacts**

10/10/99Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.4



Industry vs. Research

	Academy	Industry
Scope	One person, 1-3 years (Ph.D. project)	100's of person-years
Skills	Latest techniques Reward for <i>innovation</i>	General SWE Reward for <i>control</i>
Success	Novelty; peer review	Satisfy specifications

10/10/99

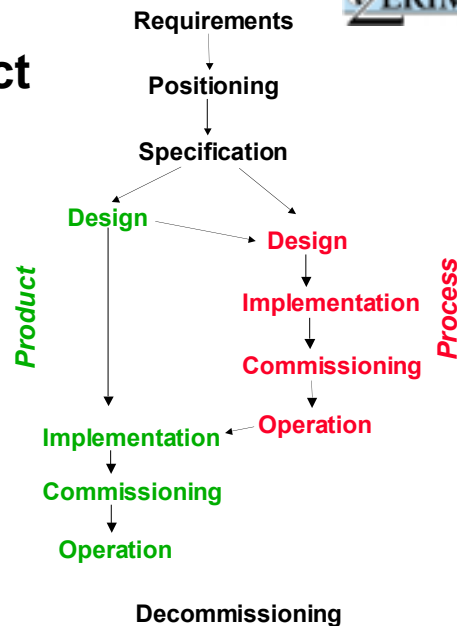
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

5



Scope: Life Cycle Project Management


- Each stage may involve different people with different skills, perspectives, and objectives.
- Artifacts provide clean interfaces between stages.



10/10/99

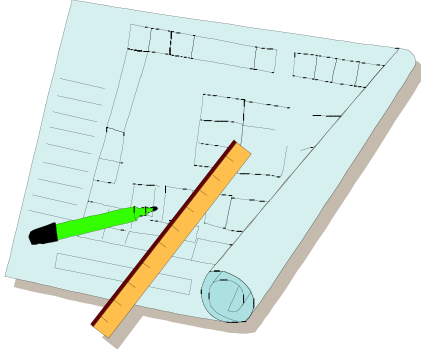
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

6




Skills:

Artifacts capture “best practice”

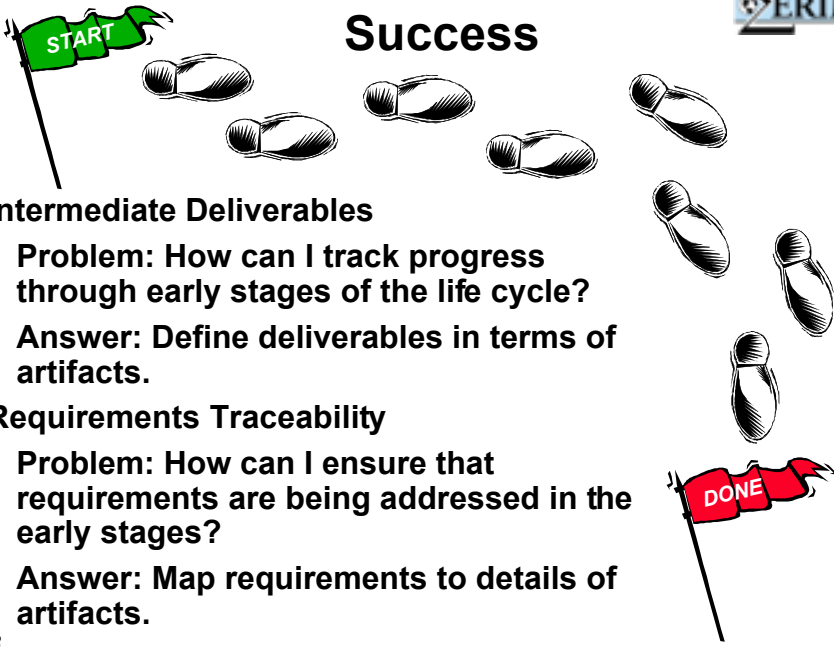


- What inputs from previous stages do I need to consider?
- What decisions do I need to make?
- How do these decisions depend on one another?

10/10/99 Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved. 7



Success



Intermediate Deliverables

- Problem: How can I track progress through early stages of the life cycle?
- Answer: Define deliverables in terms of artifacts.

Requirements Traceability

- Problem: How can I ensure that requirements are being addressed in the early stages?
- Answer: Map requirements to details of artifacts.

3+3
10/10/99 Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved. 8



Overview

- Why Engineering Artifacts?
- **OO Artifacts for MAS Development**
 - **Agent = Object++**
 - **So start with proven OO methods**
- Toward Agent-Specific Artifacts

10/10/99

Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

9



Spreading Localization

	Machine Language	Structured Programming	Object-Oriented Programming	Agent-Oriented Programming
How does a unit behave? (Code)	Diffuse	Focused	Focused	Focused
What does a unit do when it runs? (State)	External	External	Internal	Internal
When does a unit run?	External	External (called)	External (message)	Internal (rules; goals)

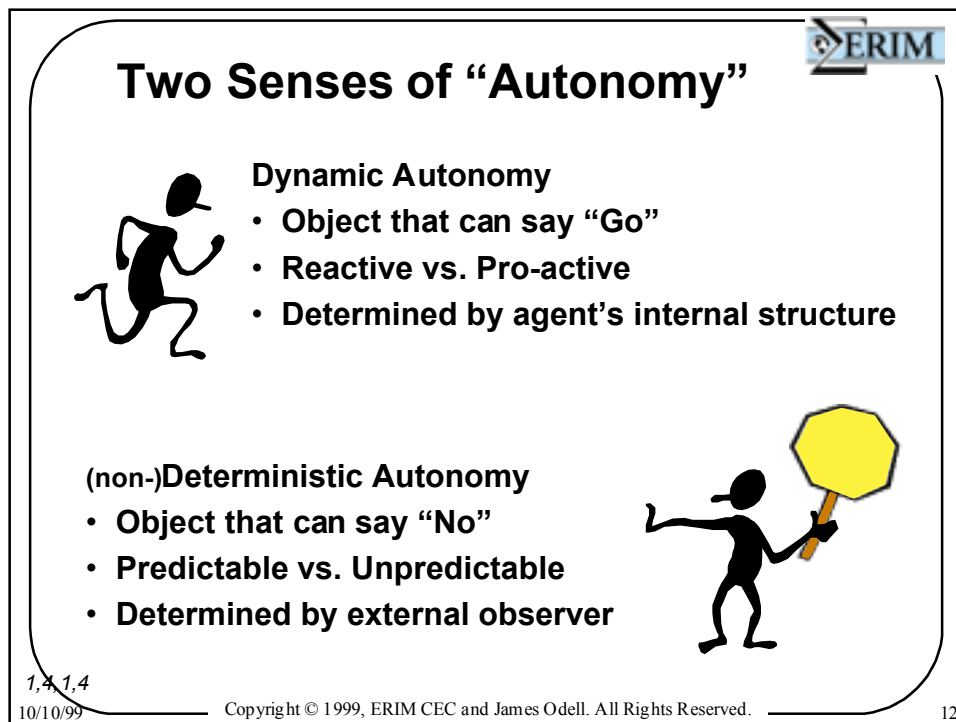
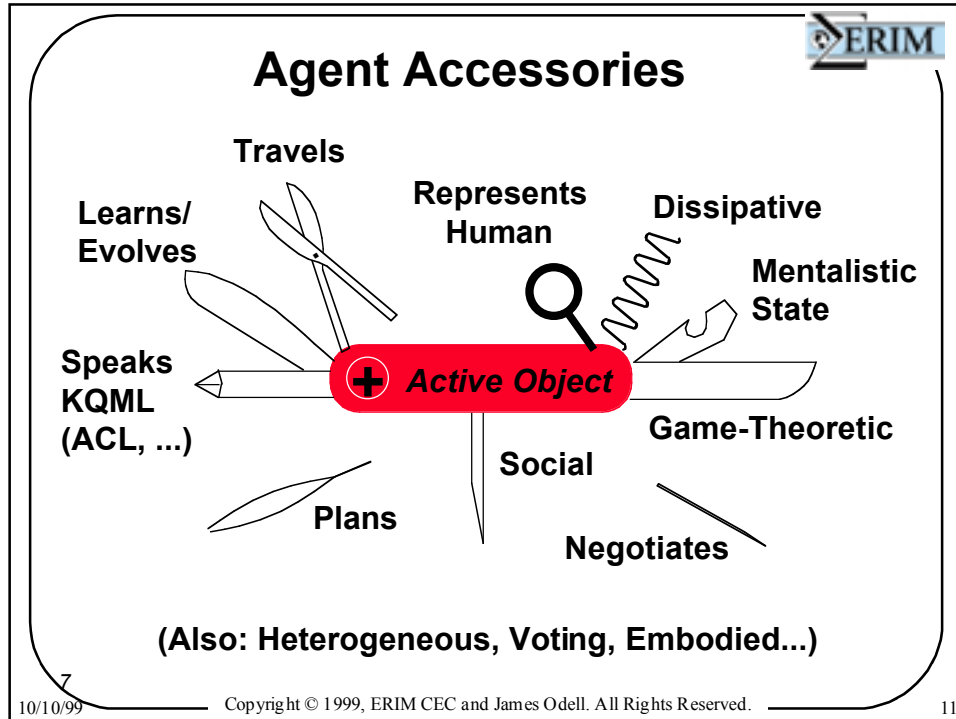
Agents are:

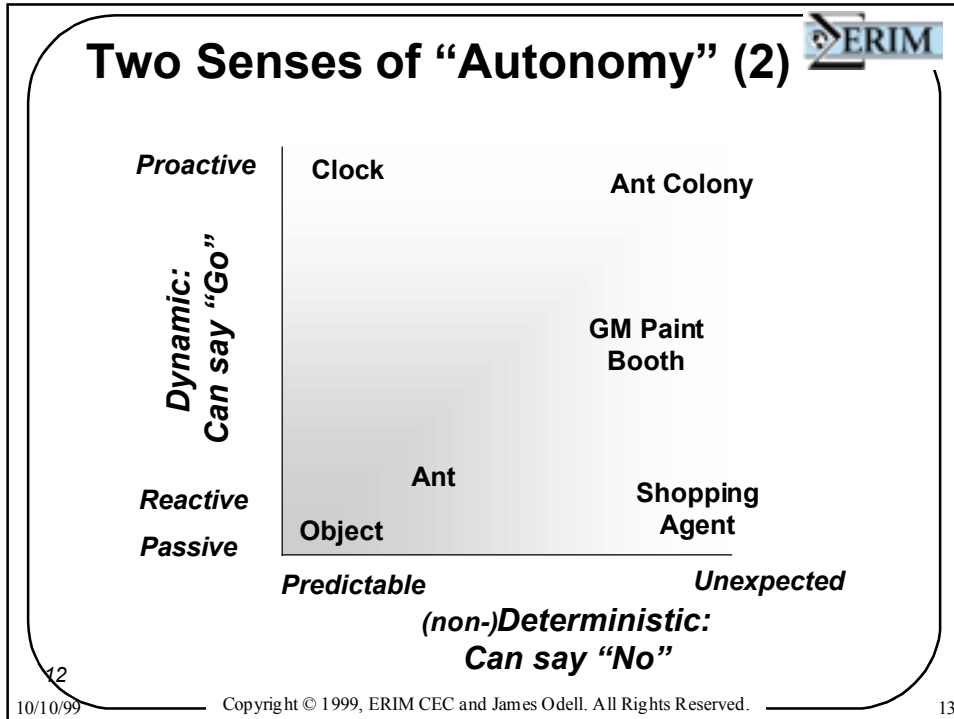
- **Objects with Initiative**
- **Objects with Attitude**
- **Pro-Active Objects**
- **Objects++**
- **Objects on Steroids**
- **Objects that can say "No" (or "Go")**

10/10/99

Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

10





Key OO Concepts in the Context of Agents

Objects	Agents
Class	Species, Roles
Message	Assertion, Query, Command, State change, Event
Attribute	State, Beliefs, Goals, Second-Order Knowledge
Method	Plans, Capabilities, Rules, Services, Responsibilities

10/10/99 Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved. 14



OO Concepts: Class

- **Class defined by static identity**
- **Roles defined by patterns of interaction**
- **Example: a company agent in a supply chain may assume different roles as**
 - Purchaser of supplies
 - Vendor of finished products
 - Employer of workers
 - Football for government regulatory agencies

Class	Role
One per object	Many per agent
Immutable	Mutable
Focus on Implementation	Focus on Function

7+1

10/10/99

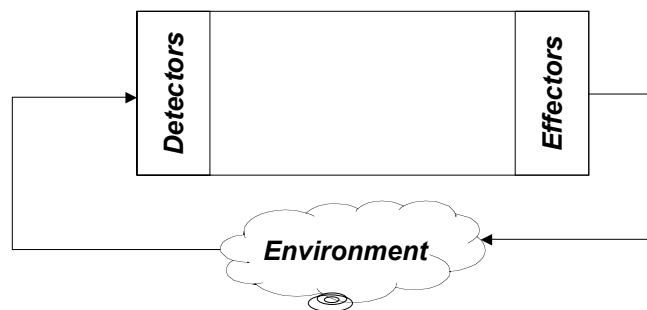
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

15



OO Concepts: Message

- **OO: <Object, method, parameters>**
- **Agent:**
 - Any change in the environment that the agent is equipped to detect.
 - At the heart of *Dynamic Autonomy*




5

10/10/99

Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

16




OO Concepts: Attributes

- **OO:**
 - Internal state variables.
 - “Variable” ↔ The mutable part of an object’s code.
- **Agents: Much more is “mutable”**
 - Beliefs about the agent and the external world
 - » Can include beliefs about agent capabilities (“methods”)
 - Desires about later states
 - Intentions (plans) for getting from here to there
 - At the heart of Deterministic Autonomy

<i>OO:</i> Attributes	<i>Agents:</i> Beliefs, Desires
↑ Rules ↓	
Methods	Intentions

3+5+1
10/10/99
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.
17




OO Concepts: Methods (1)

Subroutine < Method < Service < Responsibility

- **Method = Subroutine + Polymorphism**
- **Service = Method + Discretion (Deterministic Autonomy; “Can say No”)**
- **Responsibility includes**
 - Services (initiated by requests other agents)
 - Responses to environmental changes
 - Self-initiated activity (Dynamic Autonomy; “Can say Go”)

10/10/99
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.
18




OO Concepts: Methods (2) Agent Capabilities

Internal	What I know how to do	Psychology
External	Laws of Nature	Physics
Both	Resources at my disposal	Economics

Implemented with

- Subroutines
- Rules
- Constraints
- ...

10/10/99
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.
19



Unified Modeling Language: A Proven Language for OO Artifacts

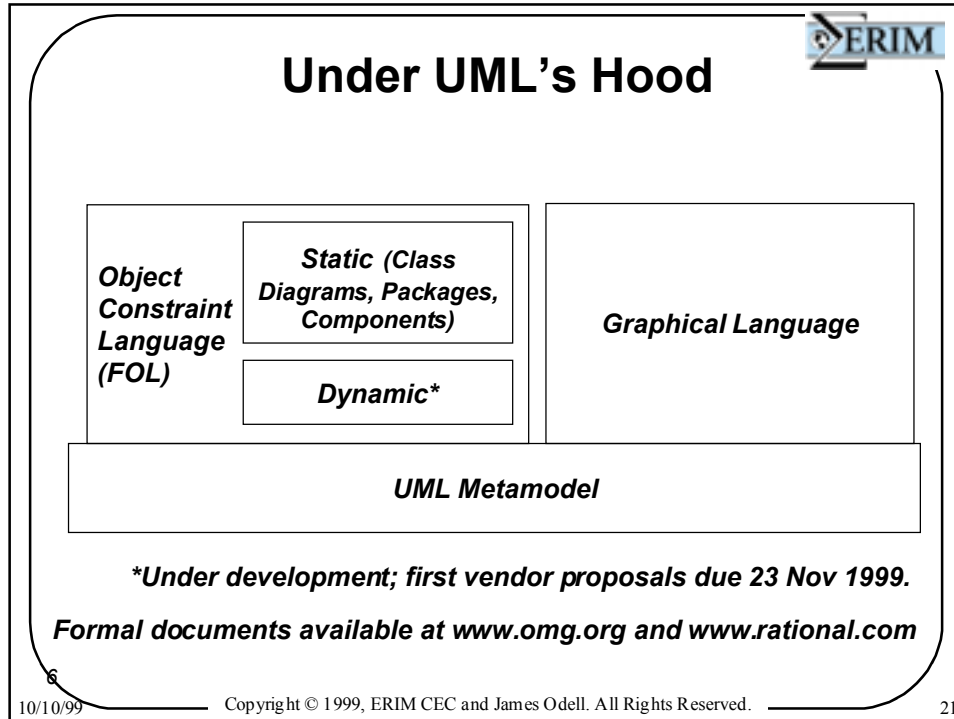
Integration and rationalization of methods promoted
by a number of OO gurus

- Booch
- Rumbaugh
- Jacobson
- Odell
- ...


Developed through OMG process

**NB: a good example of a standard *following*
technology rather than *preceding* it.**

10/10/99
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.
20



-
- ## Object Constraint Language (OCL)
- Language developed at IBM.
 - Side-effect free, constraint language used to specify well-formedness rules (static semantics).
 - The rules specify constraints over attributes and associations defined in the metamodel.
 - Basic types - Boolean, Integer, String, Set, Sequence, ...
 - Operations on the basic types:
 - and, or, not, implies, ...
 - +, -, >, =, ...
 - aSet.union (anotherSet)
 - aSet.forAll (e | Pred (e))
 - aSet.select (e | Pred (e))
 - aSequence.at (index)
 - User-defined types: the metaclasses
 - Properties: attributes, association roles, and operations
 - anInstance.property
 - Operation
 - anOp (p : ParType) : RetType;
 - anOp (p) = ... function of p ...
- 10/10/99Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.22



OCL Examples

Association

[1] The **AssociationRoles** must have a unique name within the **Association**.

self.allRoles.forAll(r1, r2 | r1.name = r2.name **implies** r1 = r2) (e.g., can't be reflexive: "spouse" can't be both association role and opposite association role for same association)

[2] At most one role in a given association may be an aggregation.

self.allRoles.select(aggregation <> none).size <= 1


Additional Operations

[3] The operation *allRoles* results in the set of all roles of the **Association**.

allRoles : Set(AssociationRole);
allRoles = self.role

5
Warmer, Jos and Anneke Kleppe, *The Object Constraint Language*, Addison-Wesley, 1999.
23

10/10/99
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.



UML Resources

- Metamodel and Object Constraint Language
- Static Diagrams**
 - Class diagrams
 - Packages
 - Component diagrams
- Dynamic Diagrams
 - Statechart
 - Collaboration diagrams
 - Sequence diagrams
 - Activity diagrams
 - Deployment diagrams
 - Use Case diagrams

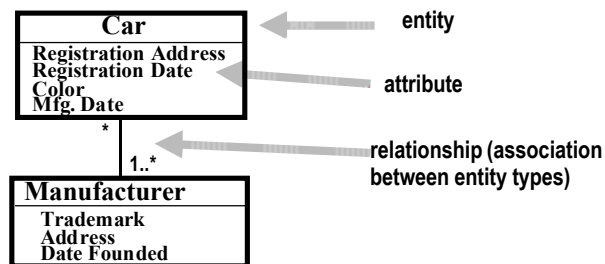
3
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.
24

10/10/99
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.



CLASS DIAGRAMS Entity-attribute-relationship approach (ERA)

- ERA was introduced to the DB community and refined by Bachman, Chen, and others.
- Relationships are usually binary (dyadic); some variants allow n-ary.
- Some variants allow relationships to have attributes.



10/10/99

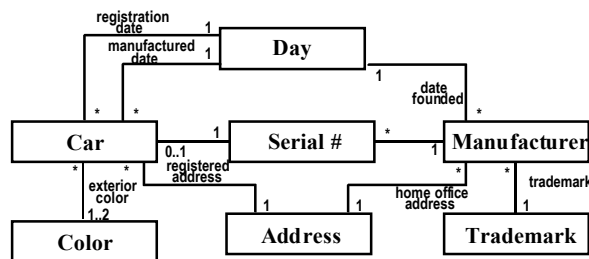
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

25



CLASS DIAGRAMS Binary relationship approach (BRA)

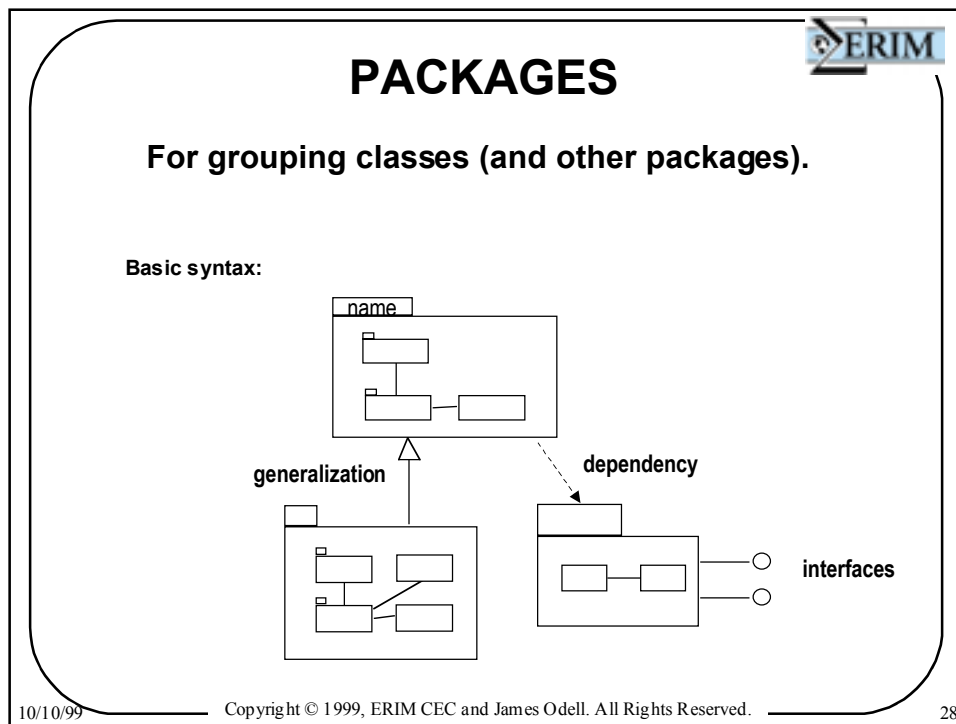
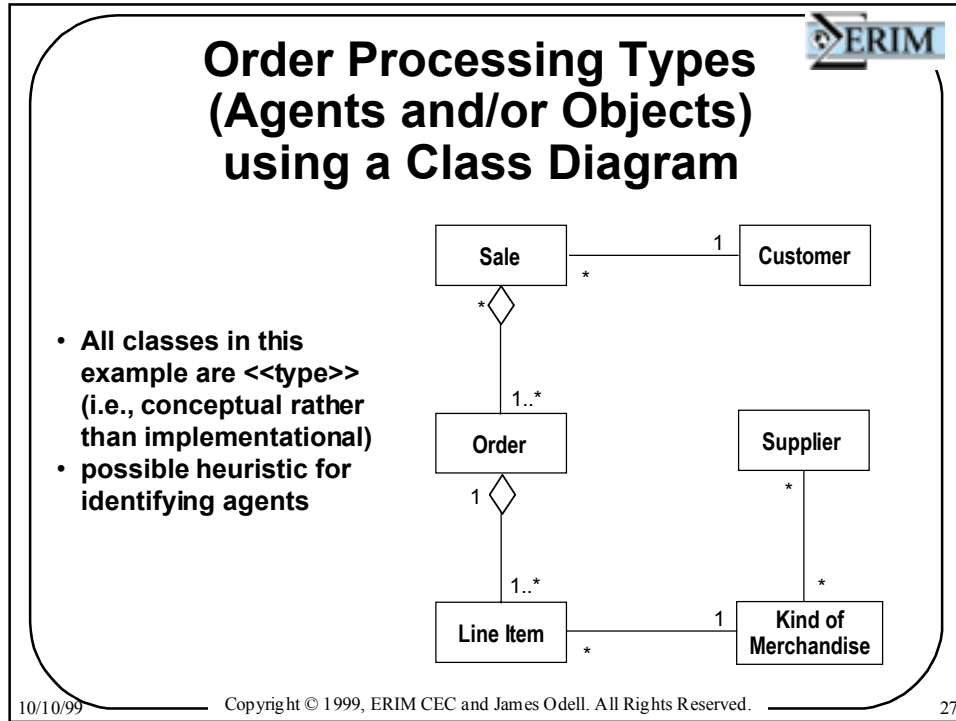
- Roots in AI and linguistics; introduced to DP community by Abrial, Bracchi, Nijssen, and Senko.
- Does not distinguish relationships from attributes.
- Relationships not restricted to binary (dyadic).
- Relationships can be component types.

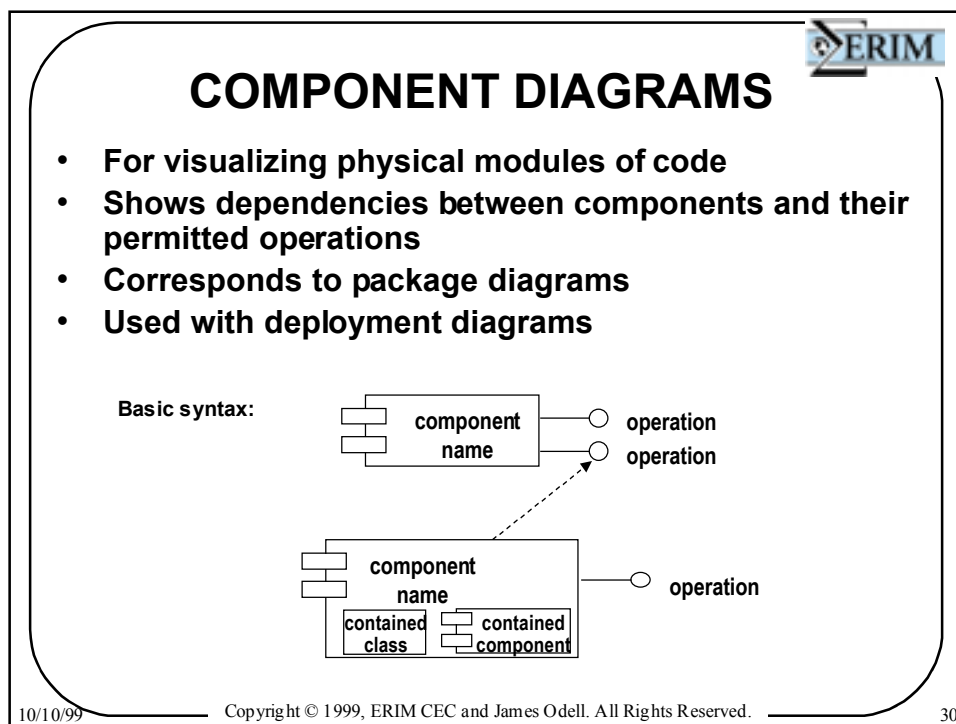
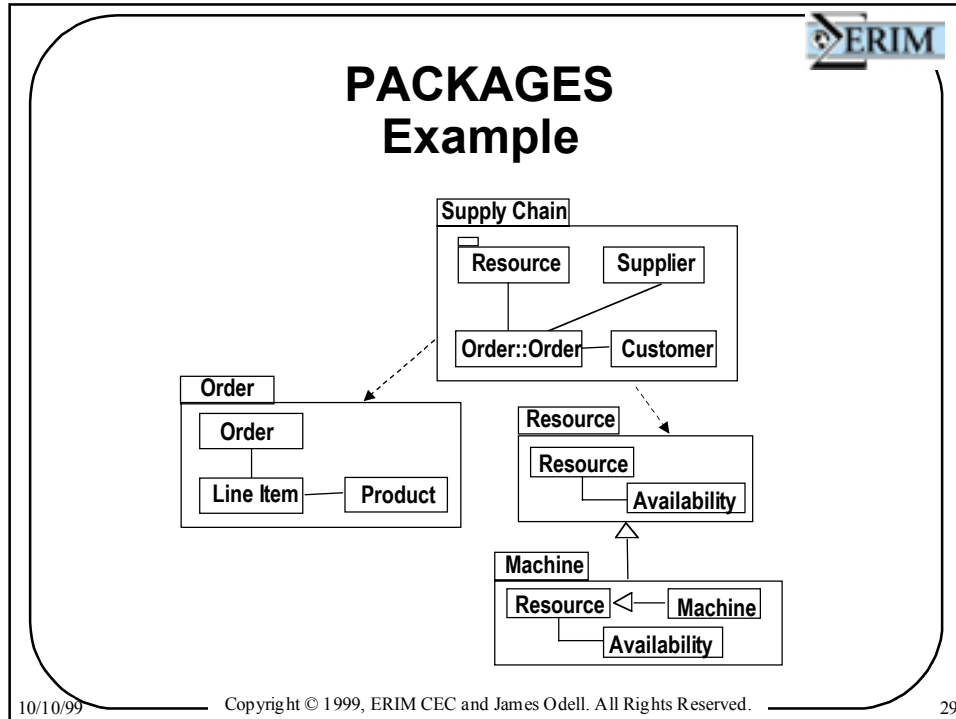


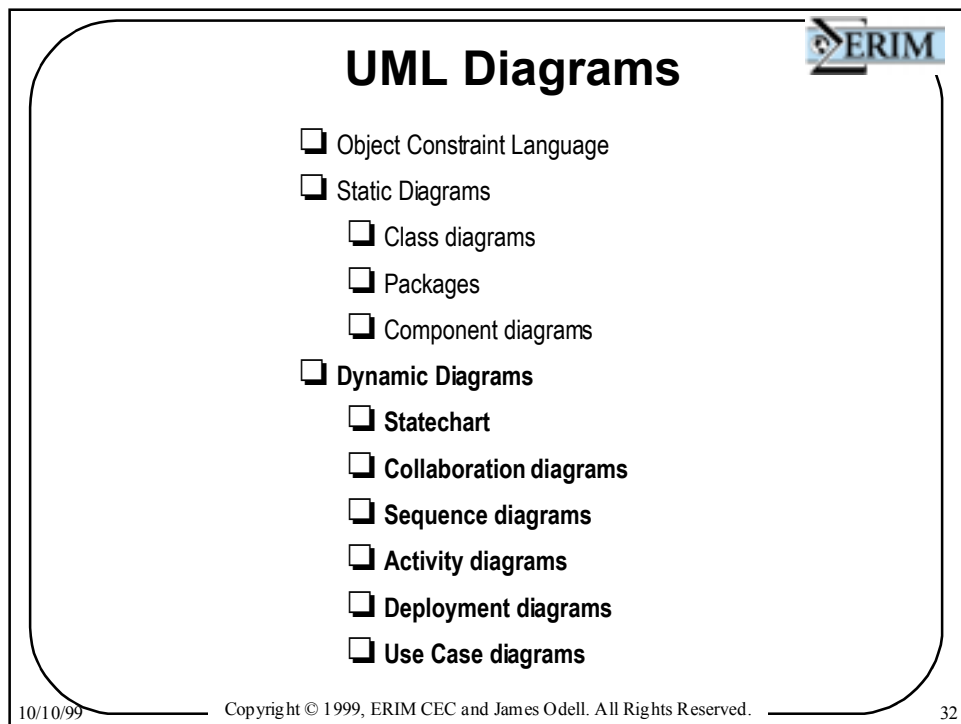
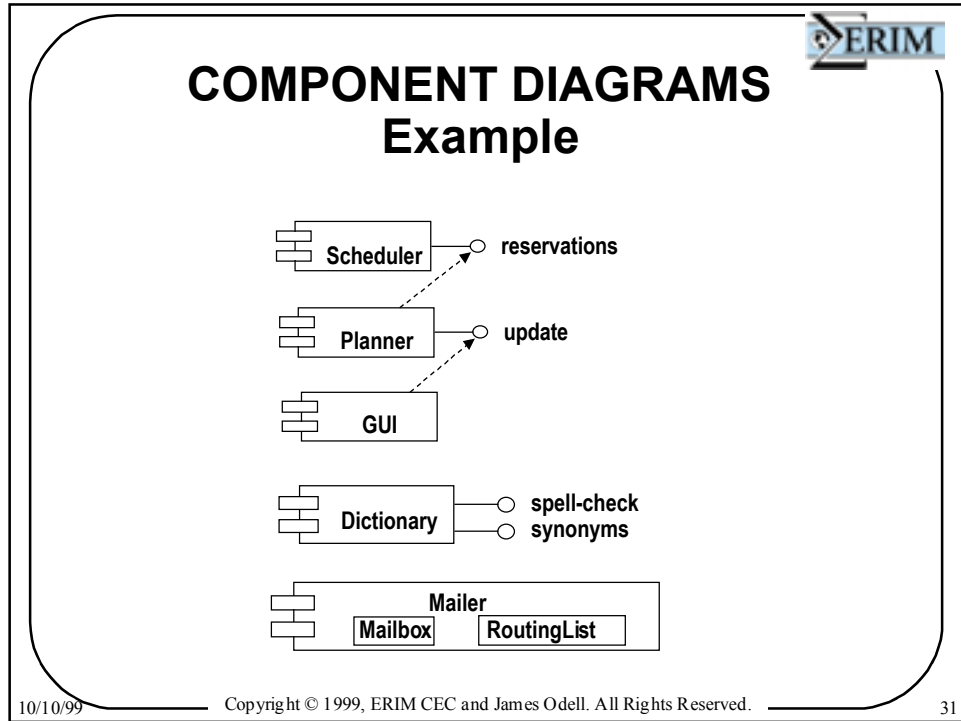
10/10/99

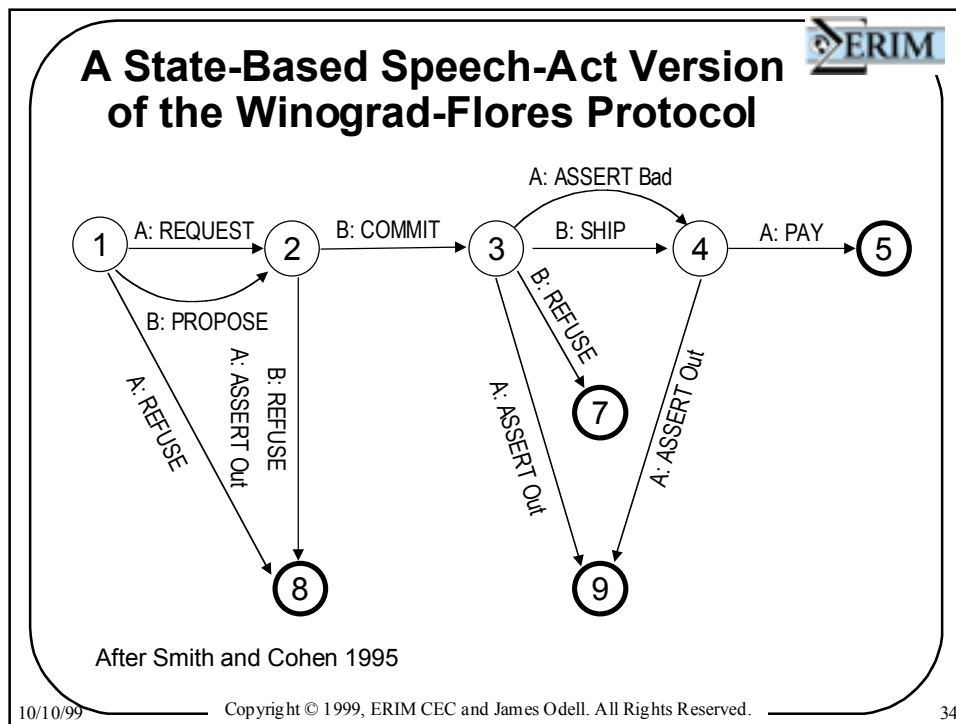
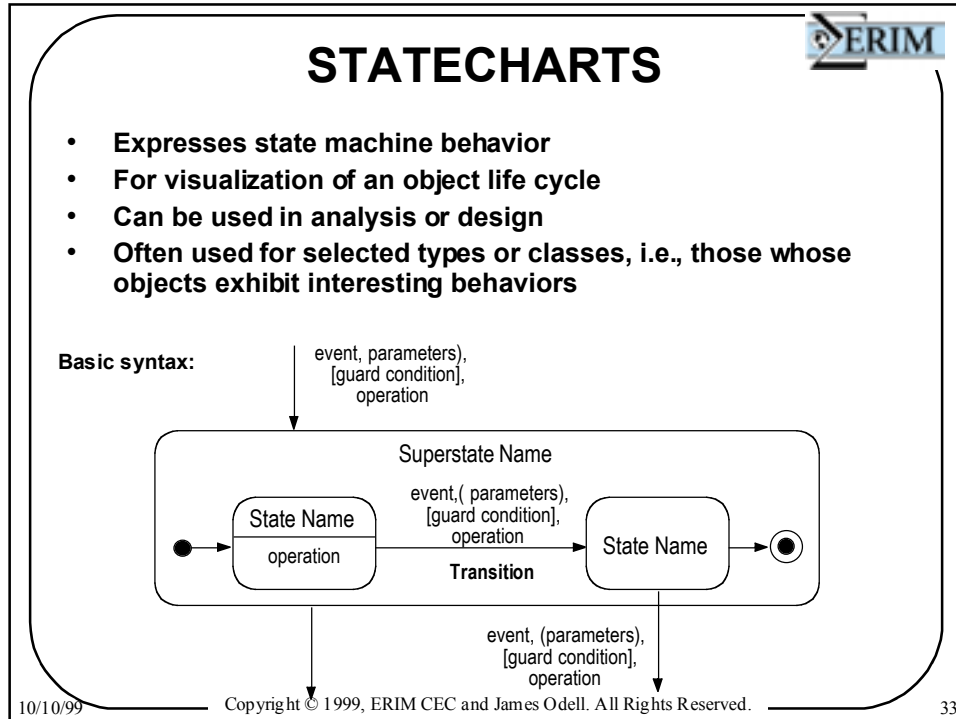
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

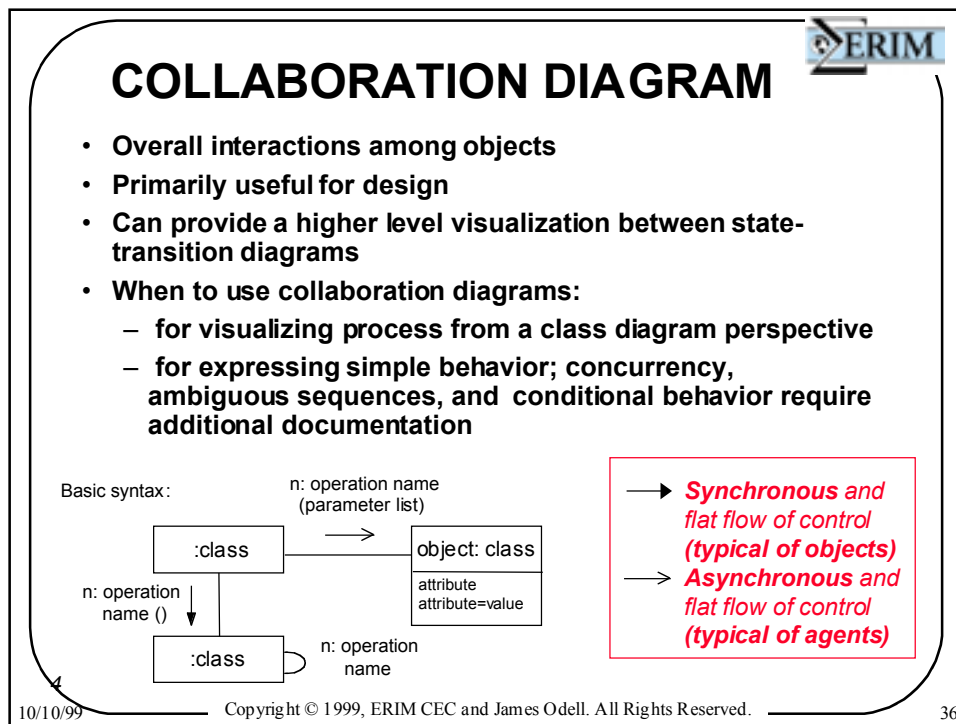
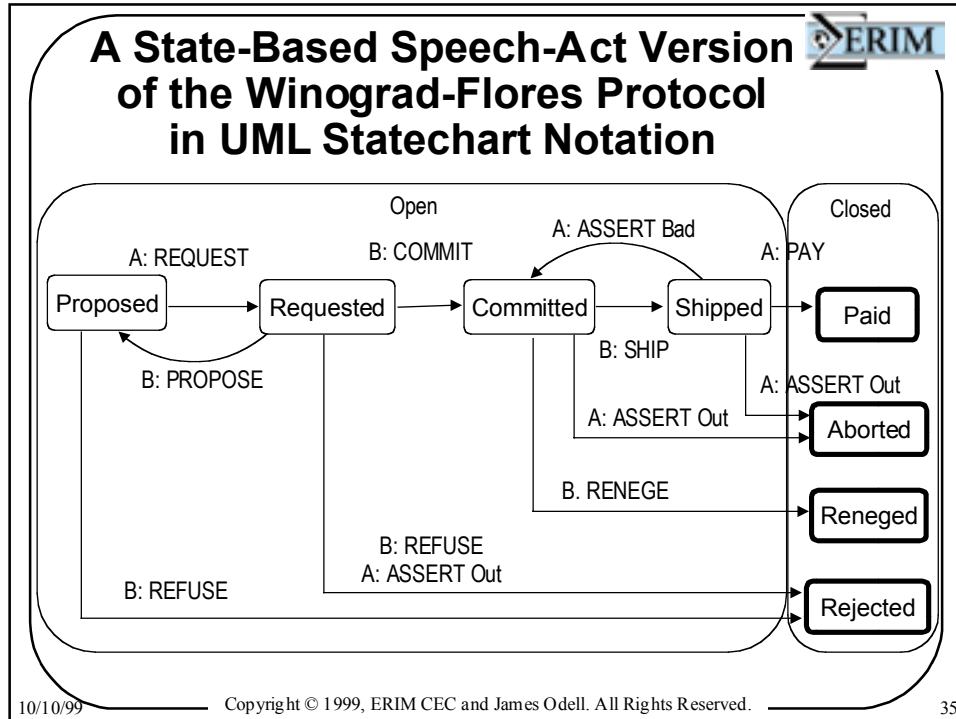
26










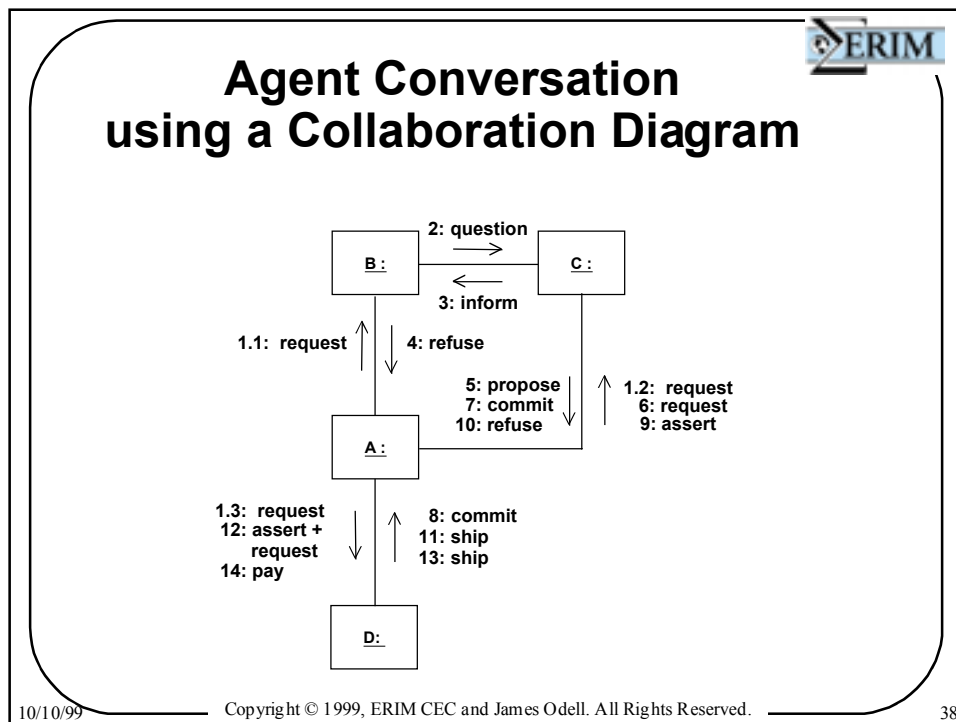


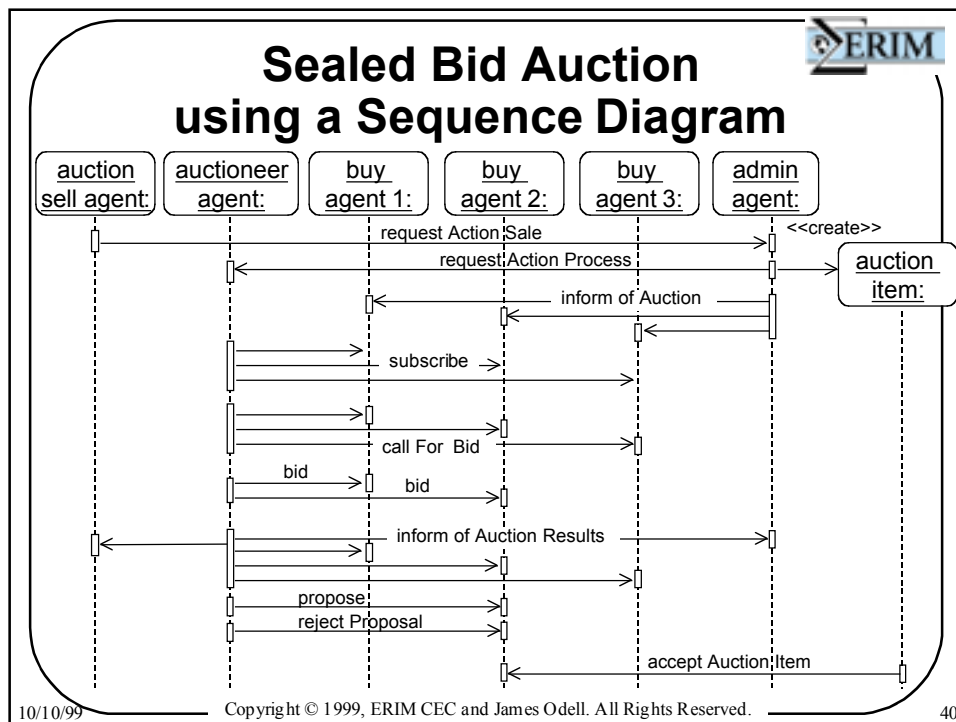
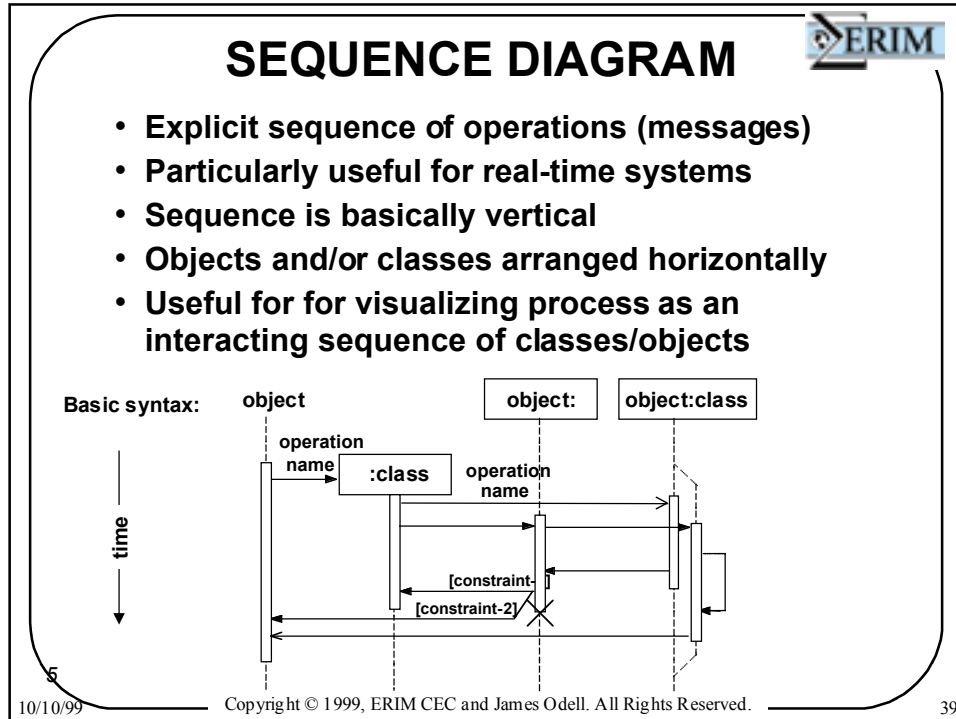
An Agent Conversation

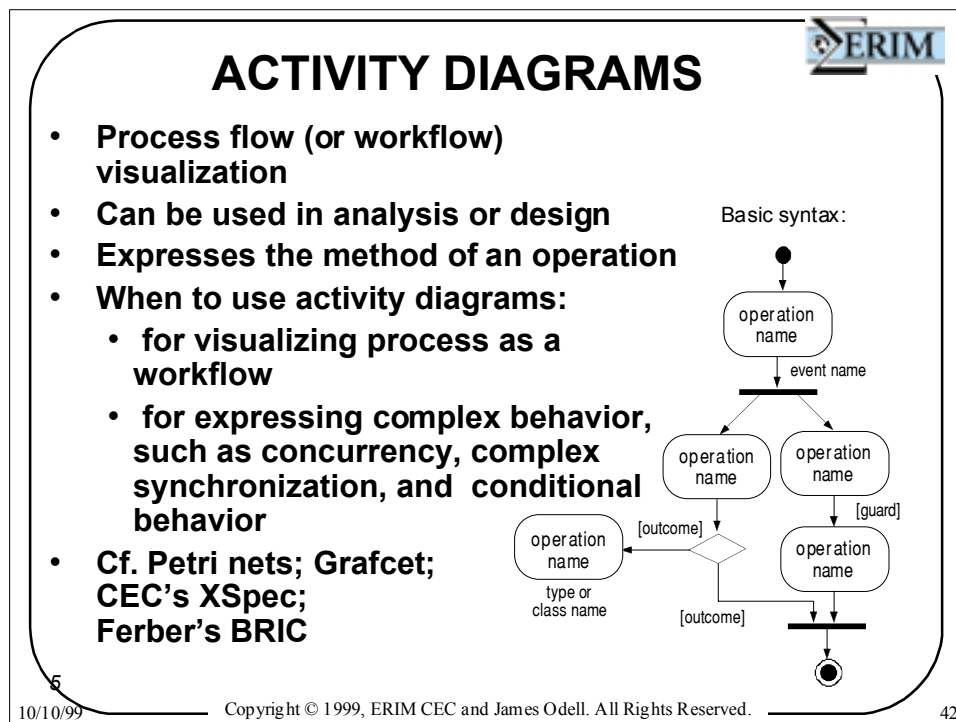
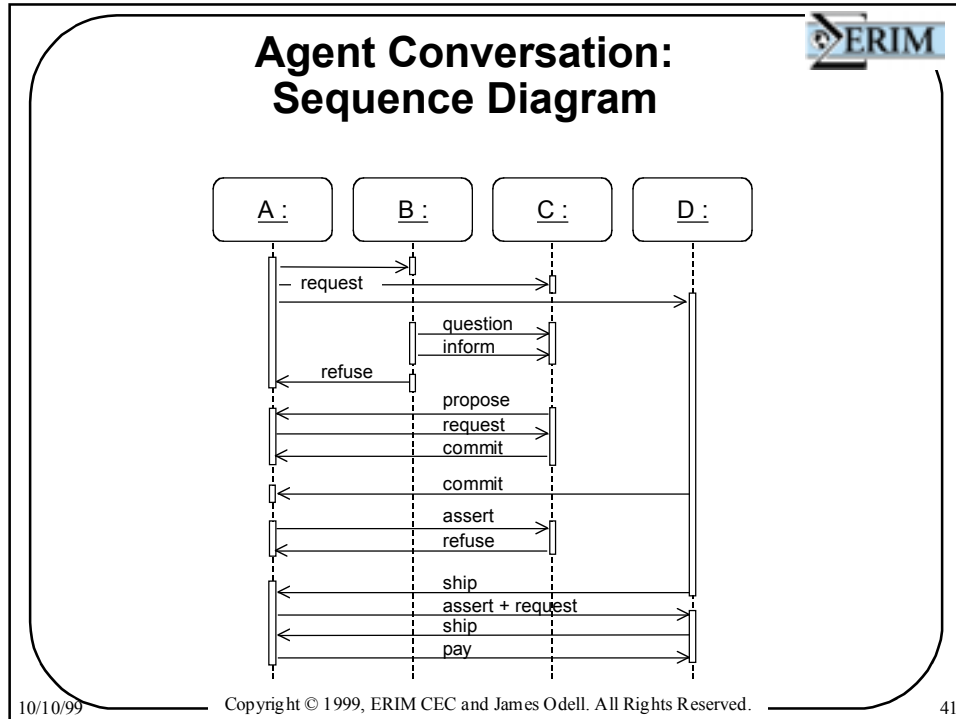


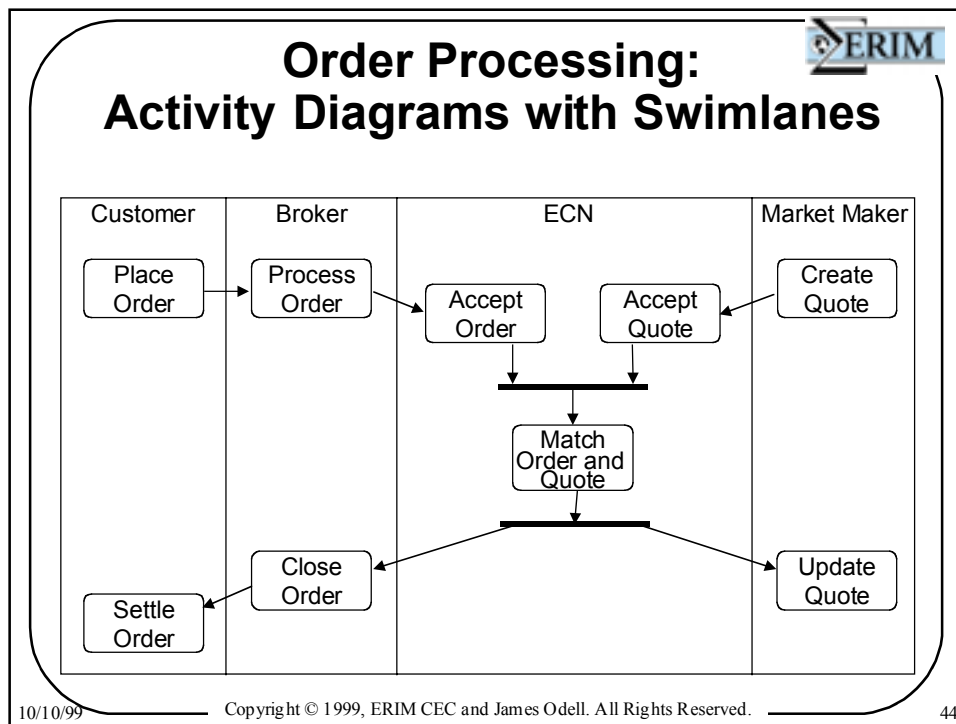
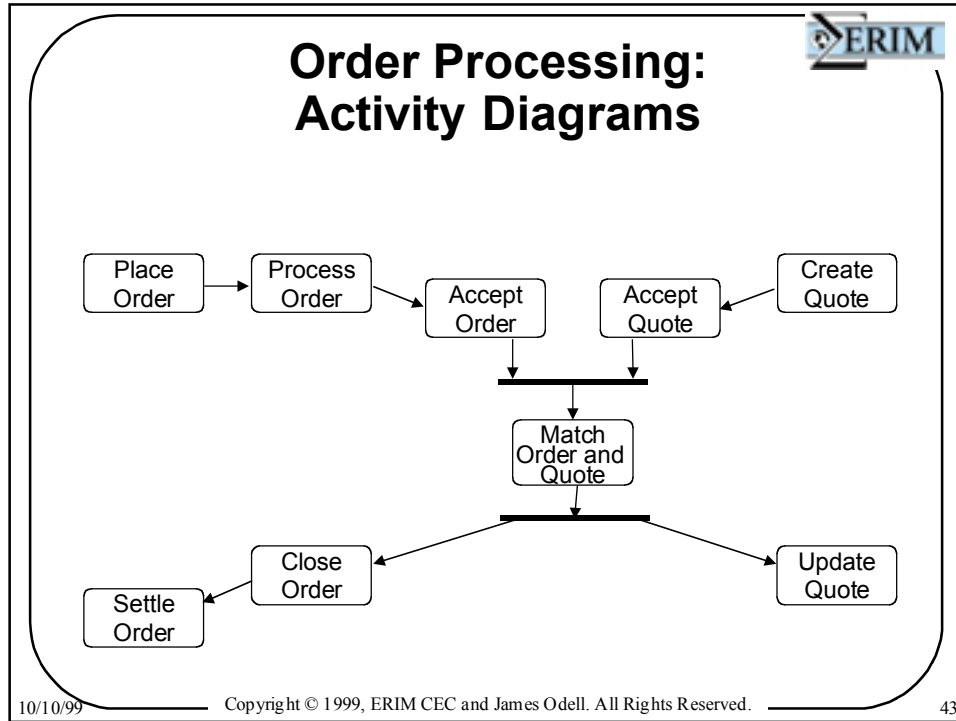
Seq	Sndr	Rcvr	Utterance	Rspnds to	Replies to	Re-solves	Completes
1.	A	B,C,D	REQUEST: Please send me 50 widgets at your catalog price by next Thursday.				
2.	B	C	QUESTION: Are you bidding on A's RFQ?	1			
3.	C	B	INFORM: Yes, I am.	2	2	2	
4.	B	A	REFUSE	3	1	1	
5.	C	A	PROPOSE (INFORM + REQUEST): How about 40 widgets at catalog price by next Friday?	1	1		
6.	A	C	REQUEST: Please send me 40 widgets at catalog price by next Friday.	5	5	5	
7.	C	A	COMMIT: I plan to send you 40 widgets at catalog price by next Friday.	6	6	6	
8.	D	A	COMMIT: I plan to send you 50 widgets at catalog price by next Thursday.	1	1	1	
9.	A	C	ASSERT: I've found a better supplier, and am not relying on your COMMIT.	7,8	7		
10.	C	A	REFUSE: I'm abandoning my COMMIT.	9	9		7
11.	D	A	SHIP: Here are your widgets. Please pay me.	1	1		8
12.	A	D	ASSERT + REQUEST: You're five short. Please send the difference.	11	11		
13.	D	A	SHIP: Here are five more widgets. Please pay me.	12	12	12	
14.	A	D	PAY	13	13	13	

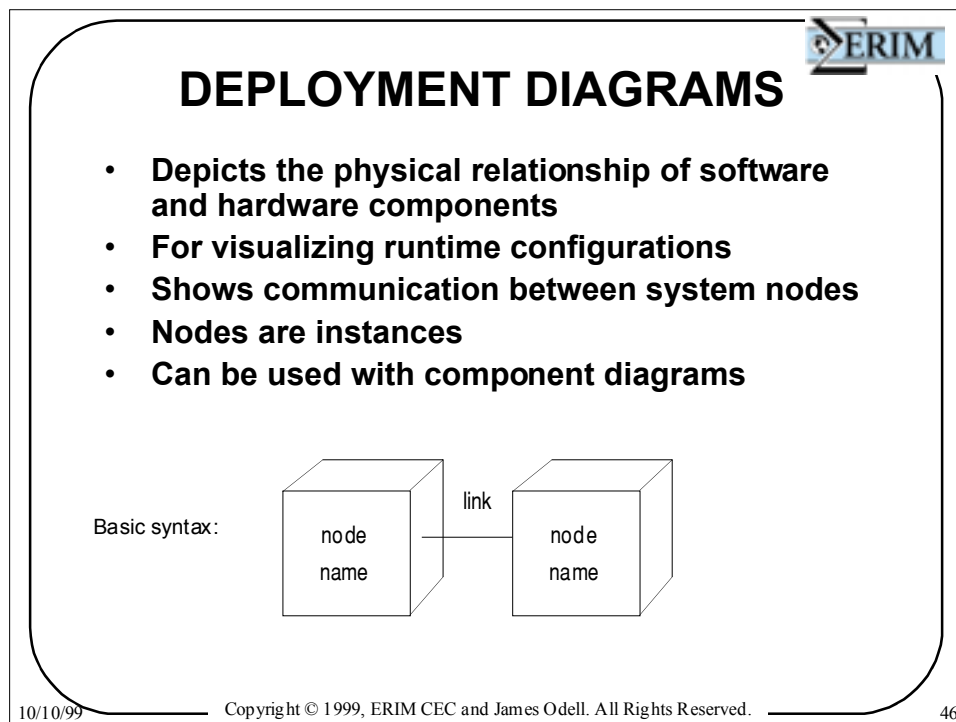
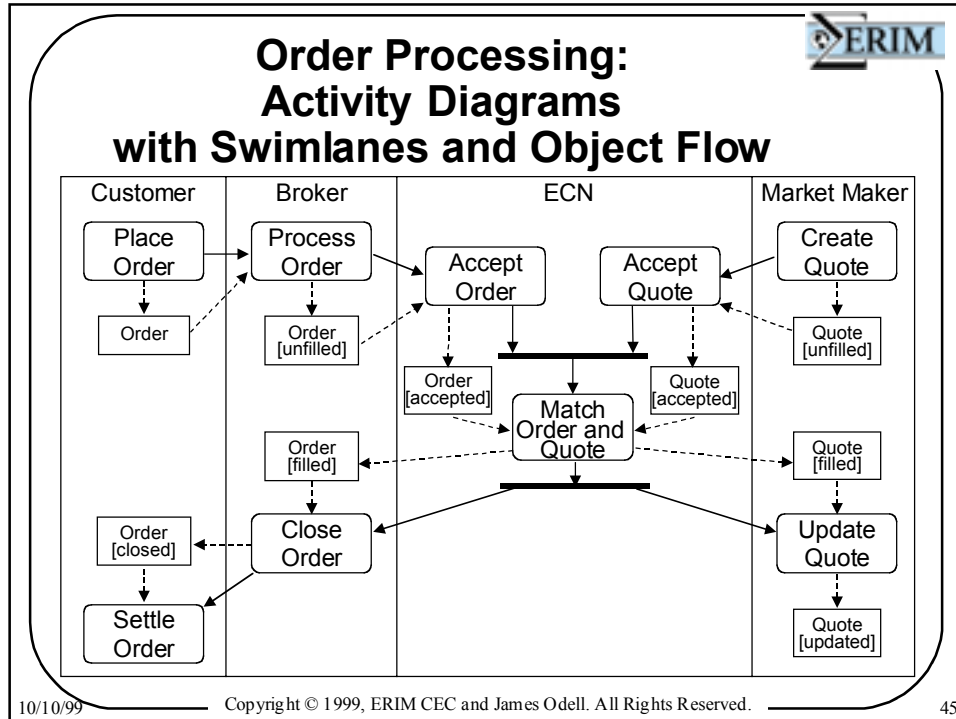
10/10/99 Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved. 37





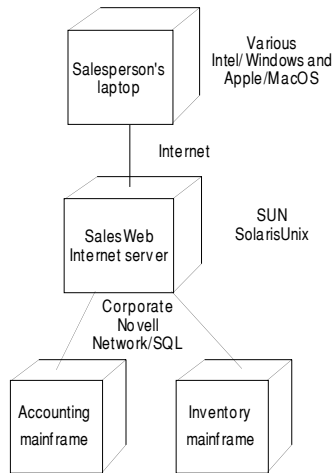








DEPLOYMENT DIAGRAMS Example



10/10/99

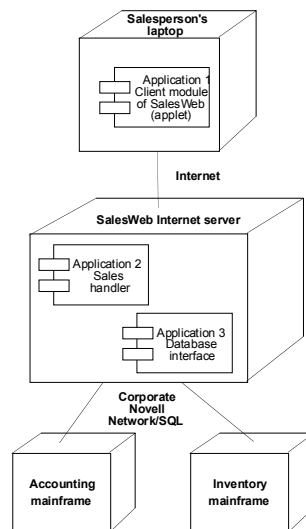
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

47



DEPLOYMENT DIAGRAMS WITH COMPONENTS

Example:



10/10/99

Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

48

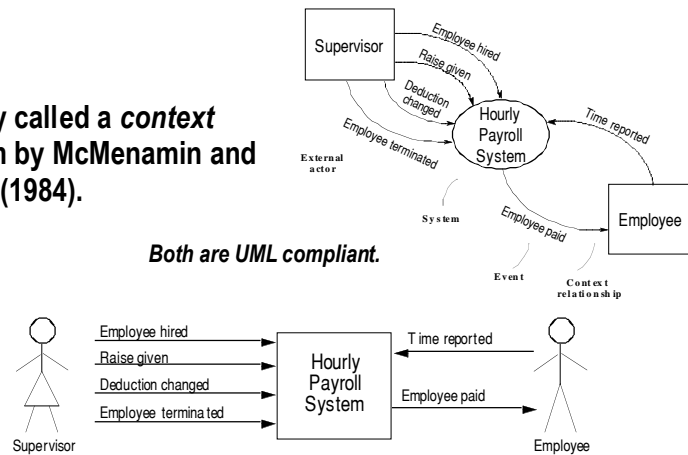


USE CASE DIAGRAMS for context specification

Expresses the relationship between a given system and its environment.

Originally called a *context diagram* by McMenamin and Palmer (1984).

Both are UML compliant.



10/10/99

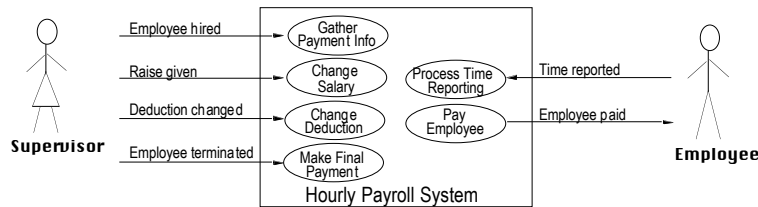
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

49



USE CASE DIAGRAMS

Shows the relationships among actors and use cases within a system.



A context diagram expressing the operations that will provide the interface between the system and its environment. UML refers to these operations as *use cases* (discussed in the next section).


10/10/99

Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

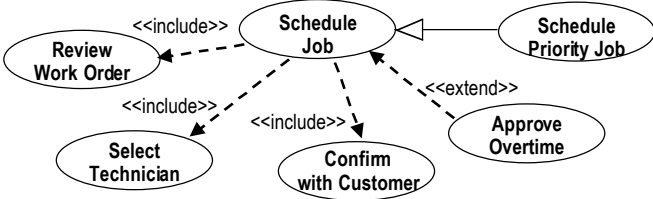
50

USE CASE DIAGRAMS

Use Case Relations



- **Includes Relation** – A “base” use case incorporates the behavior of another use case, i.e., mandatory.
- **Extends Relation** – A “base” use case adds to the behavior specified in another use case, i.e., optional.
- **Generalization**– A “base” use case that has more specialized use cases.




“Uses” has been replaced by include and generalization.

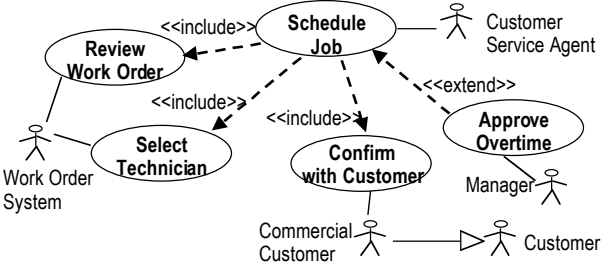
10/10/99 Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved. 51

USE CASE DIAGRAMS

with actors



Actors are roles played when interacting with the use case. They may be human, hardware, or software.



- The symbol can be confusing: *an actor does not have to be a person.*
- It represents the external interfaces to the system.
- Identifying the actor helps the team in understanding the stakeholder of a use case.
- Since actors are types, they may be subtyped.

10/10/99 Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved. 52



Overview

- Why Engineering Artifacts?
- OO Artifacts for MAS Development
- **Toward Agent-Specific Artifacts**

10/10/99

Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

53



Some Deficiencies of UML Diagrams for Agents

UML has no “off-the-shelf” constructs for:

- mobility
- role changes
- generative functions, such as cloning, birthing, reproduction
- parasitism and symbiosis
- agents *on* “cell membranes” versus *within* membranes
- emergent phenomena
- ...

10/10/99

Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

54



UML Extension Mechanisms

- Labels in <<...>> (“stereotypes”)
- Extensions may become normative in later versions of the language.
- These are only suggestions; help us develop the right notations.
 - What do we need to represent?
 - What are the most helpful ways to represent them?

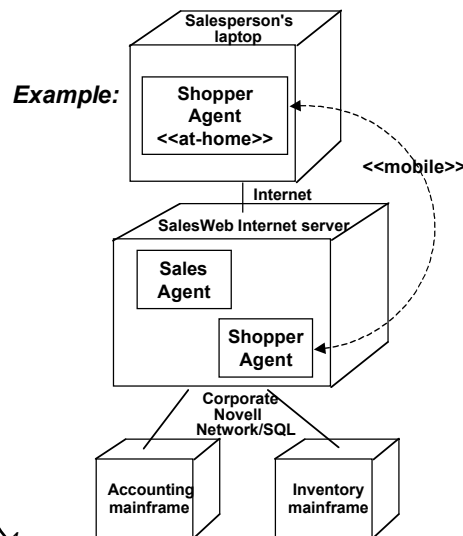
10/10/99

Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

55



DEPLOYMENT DIAGRAMS Extending UML for Mobile Agents



Innovations:

- Class (not just component) in deployment box
- <<at-home>> to indicate origin
- <<mobile>> link to show possible movement

10/10/99

Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

56



Application-oriented view

- How to define agents?
 - Class diagrams
 - Activity diagrams via swimlanes
 - Sequence diagrams defined first at the agent level, then abstracted to the role level
- How to derive roles?
 - Dooley analysis from collaboration diagram
 - Time analysis from sequence diagram

10/10/99

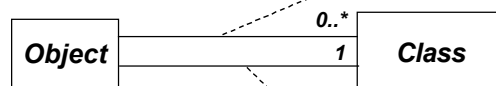
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

57



Multi-Role Agents

1. Defining Roles in the UML Metamodel:



Other roles the object can fill

Primary role of the object

2. Detecting Roles:

- From Dooley analysis
- From sequence diagrams

3. Representing Roles

2+3+1

10/10/99

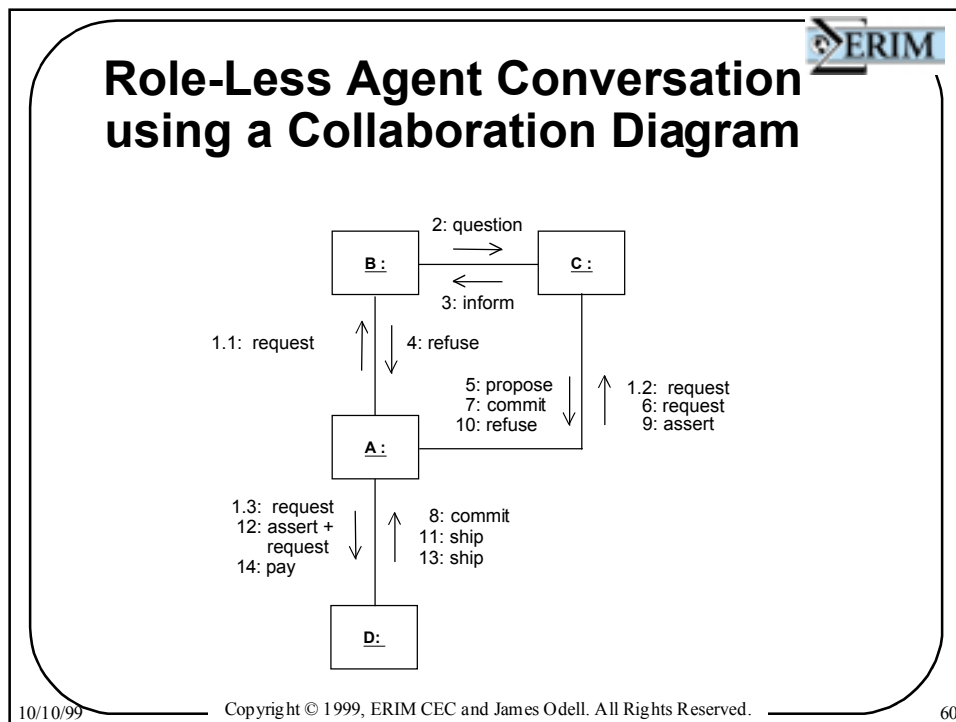
Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

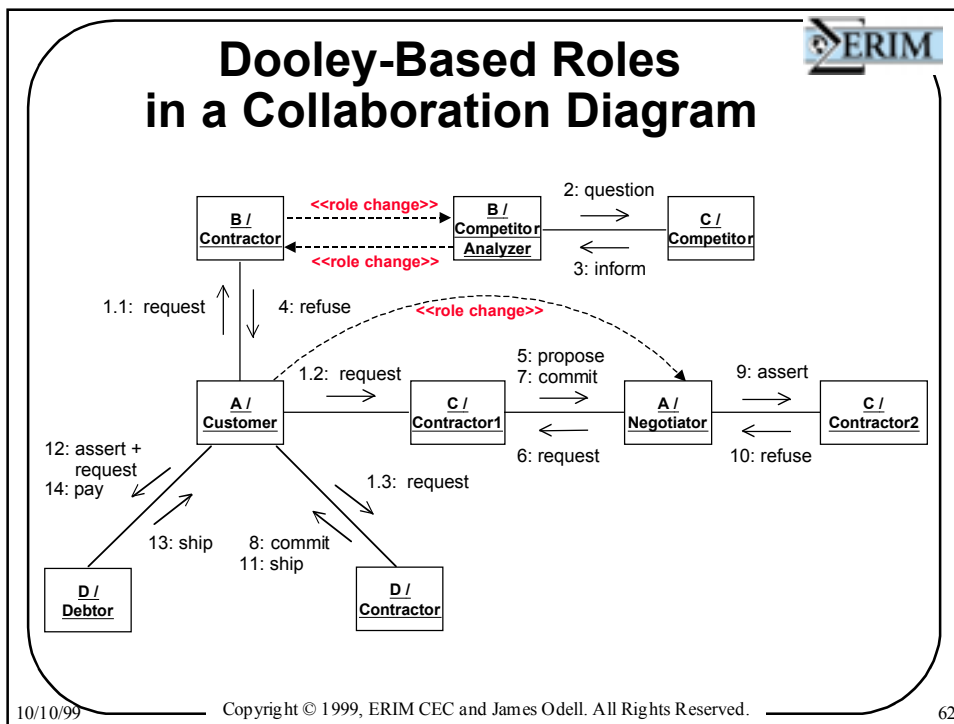
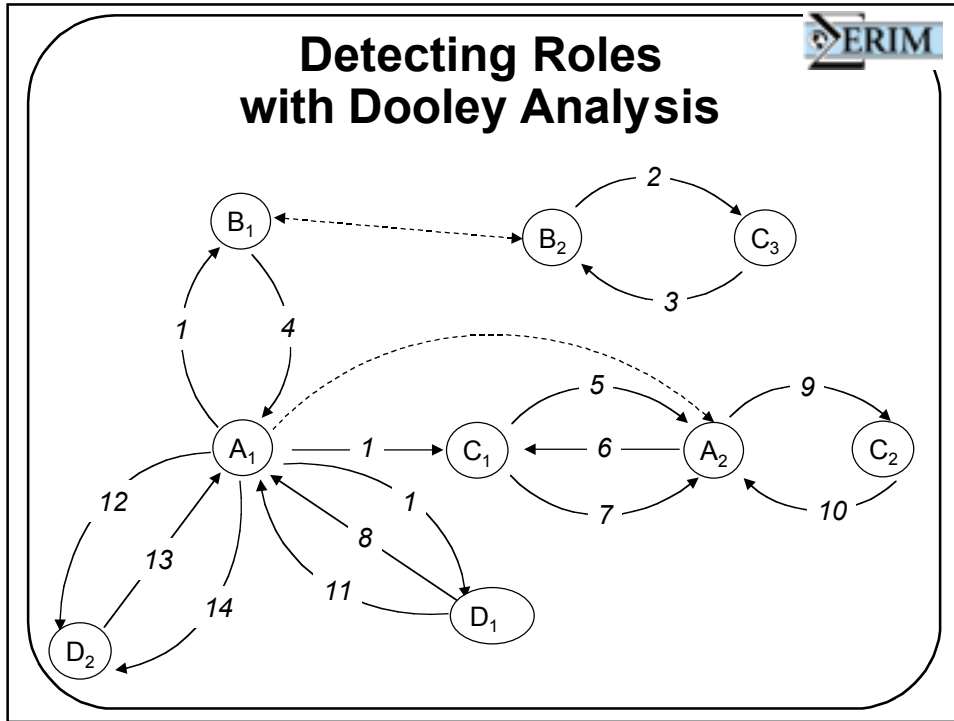
58

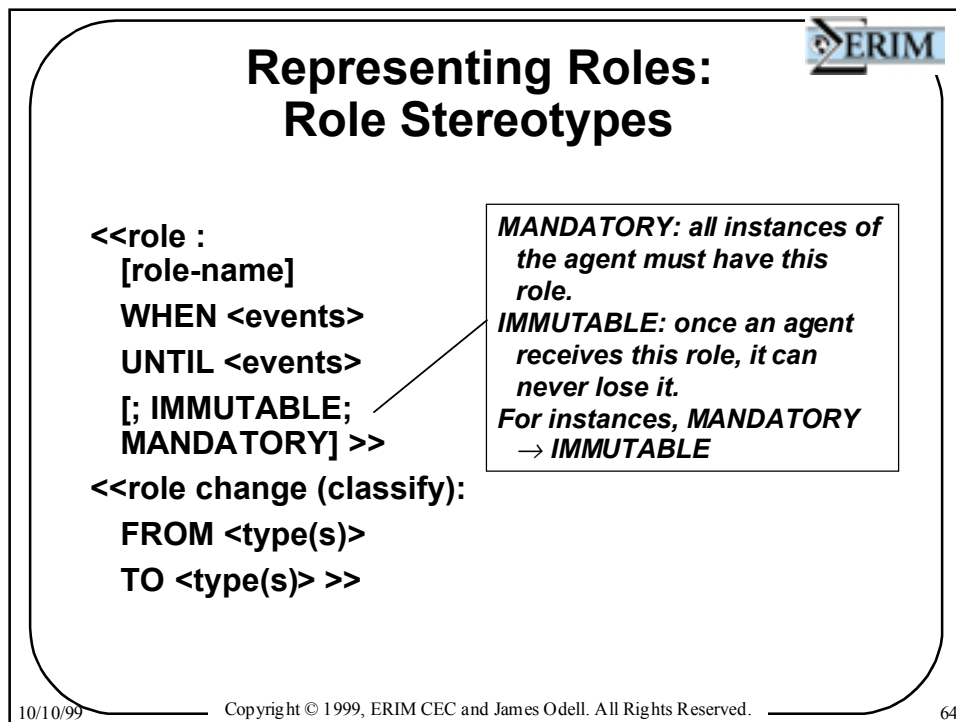
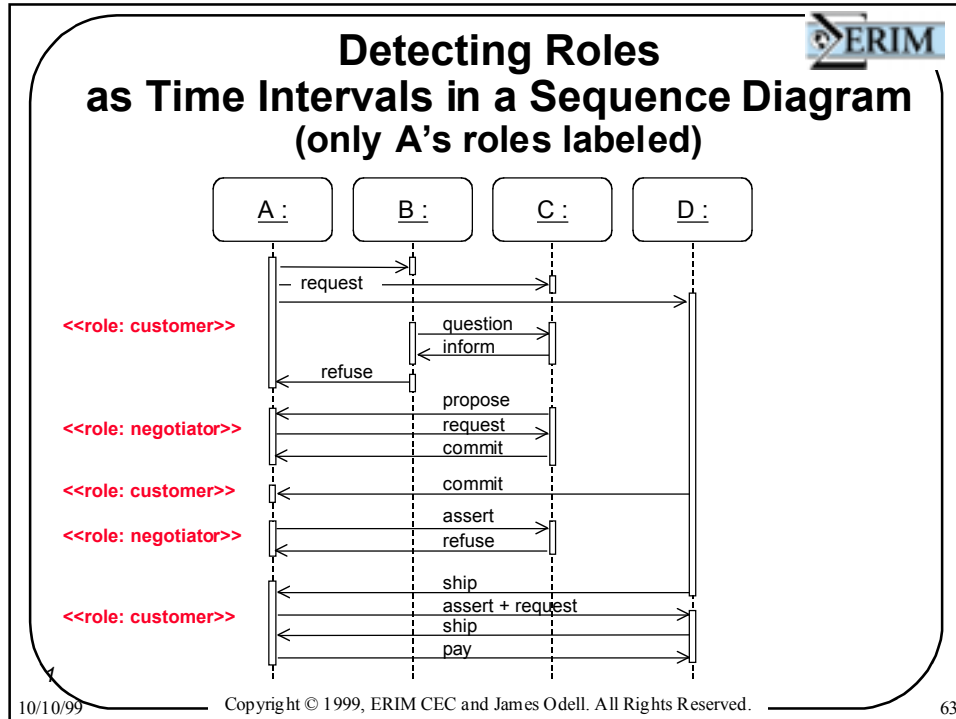
An Agent Conversation

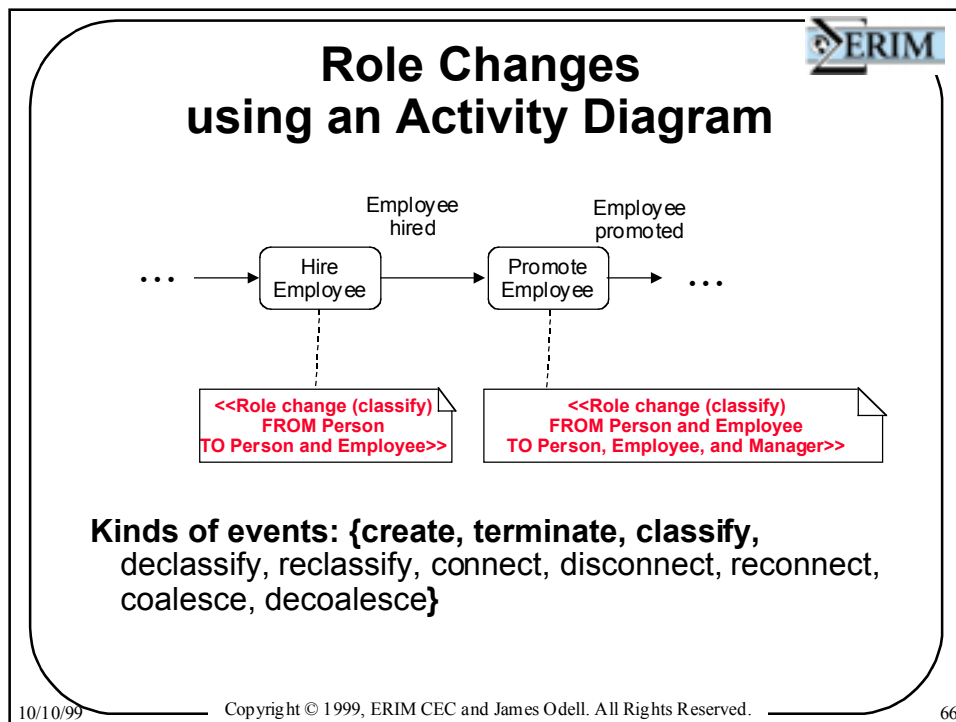
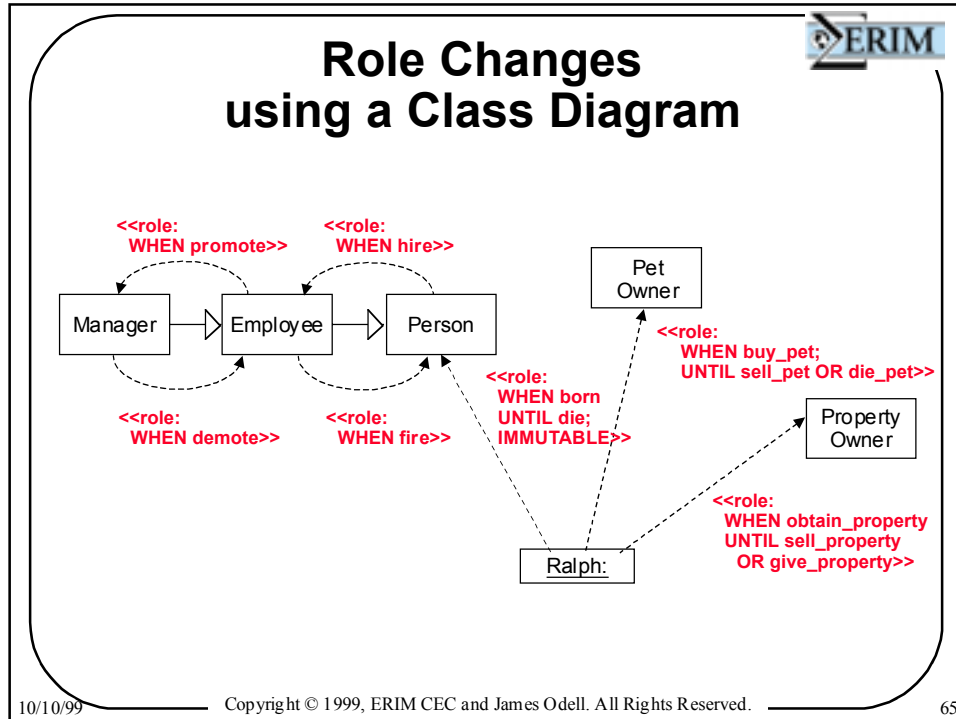
Seq	Sndr	Rcvr	Utterance	Rspnds to	Replies to	Re-solves	Com-pletes
1.	A	B,C,D	REQUEST: Please send me 50 widgets at your catalog price by next Thursday.				
2.	B	C	QUESTION: Are you bidding on A's RFQ?	1			
3.	C	B	INFORM: Yes, I am.	2	2	2	
4.	B	A	REFUSE	3	1	1	
5.	C	A	PROPOSE (INFORM + REQUEST): How about 40 widgets at catalog price by next Friday?	1	1		
6.	A	C	REQUEST: Please send me 40 widgets at catalog price by next Friday.	5	5	5	
7.	C	A	COMMIT: I plan to send you 40 widgets at catalog price by next Friday.	6	6	6	
8.	D	A	COMMIT: I plan to send you 50 widgets at catalog price by next Thursday.	1	1	1	
9.	A	C	ASSERT: I've found a better supplier, and am not relying on your COMMIT.	7,8	7		
10.	C	A	REFUSE: I'm abandoning my COMMIT.	9	9		7
11.	D	A	SHIP: Here are your widgets. Please pay me.	1	1		8
12.	A	D	ASSERT + REQUEST: You're five short. Please send the difference.	11	11		
13.	D	A	SHIP: Here are five more widgets. Please pay me.	12	12	12	
14.	A	D	PAY	13	13	13	

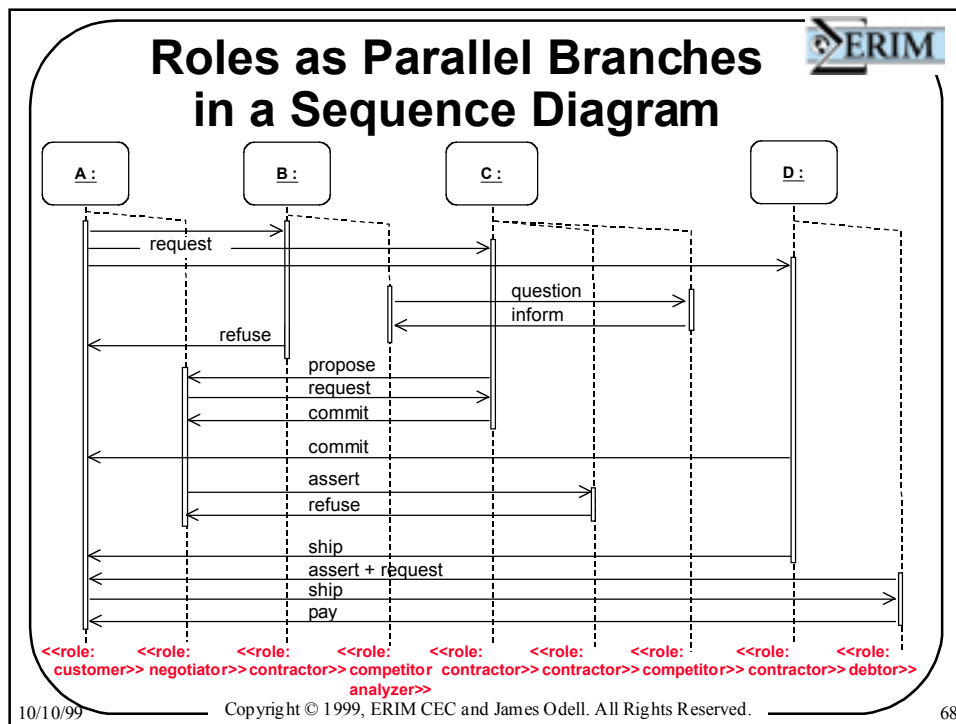
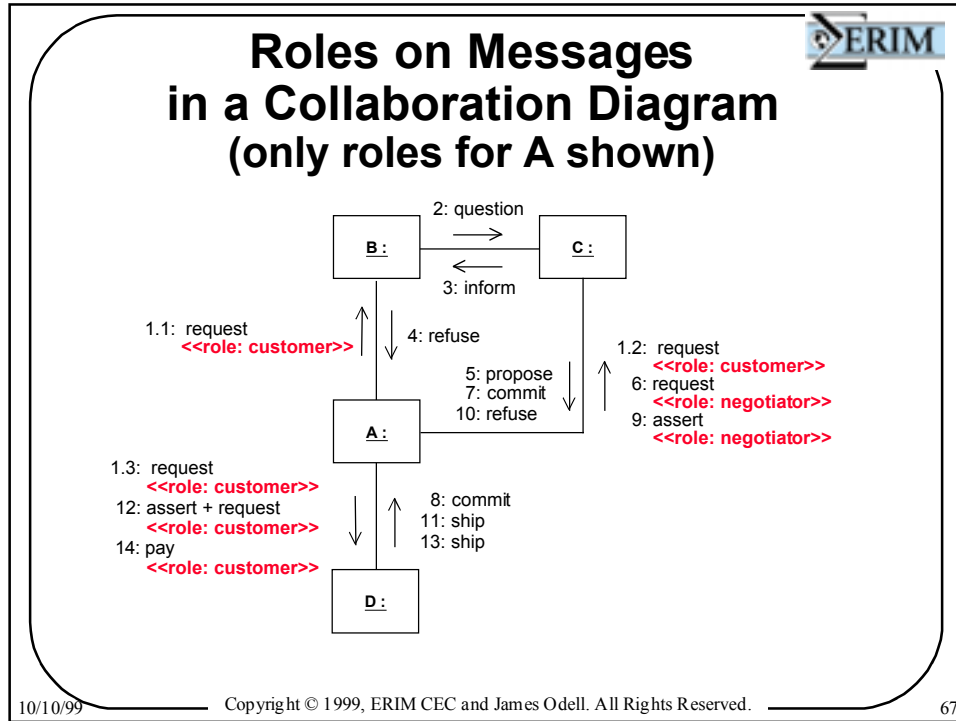
10/10/99 Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved. 59

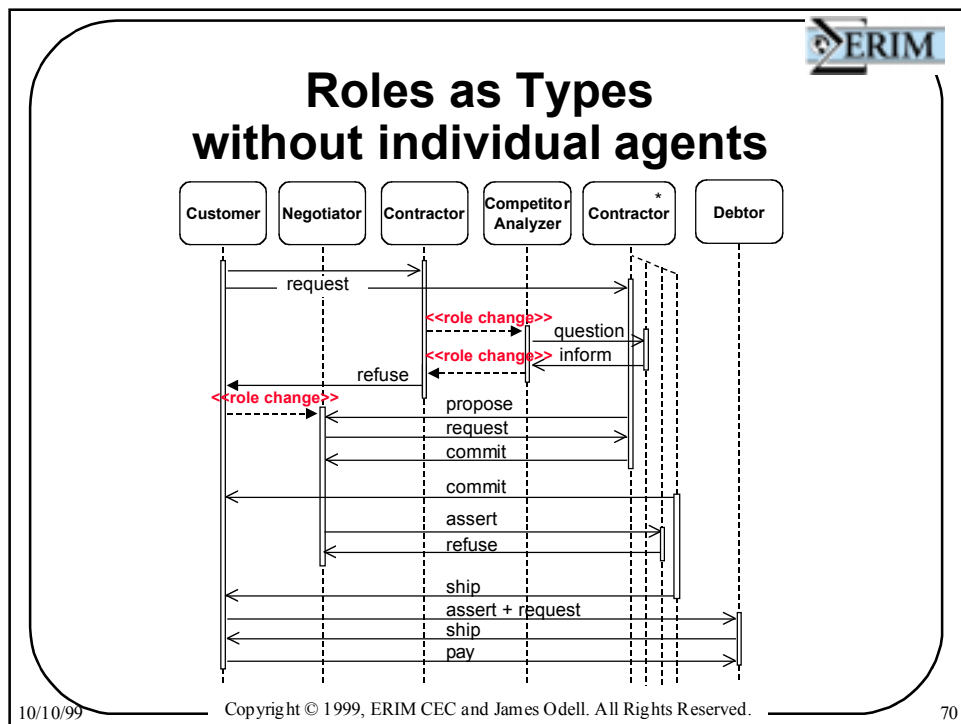
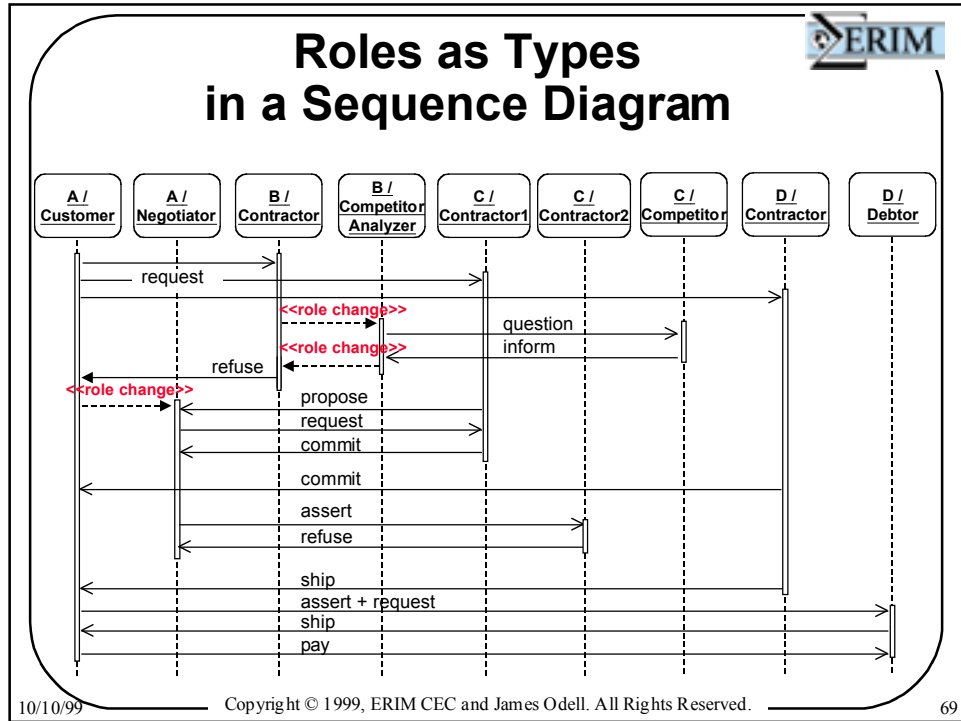


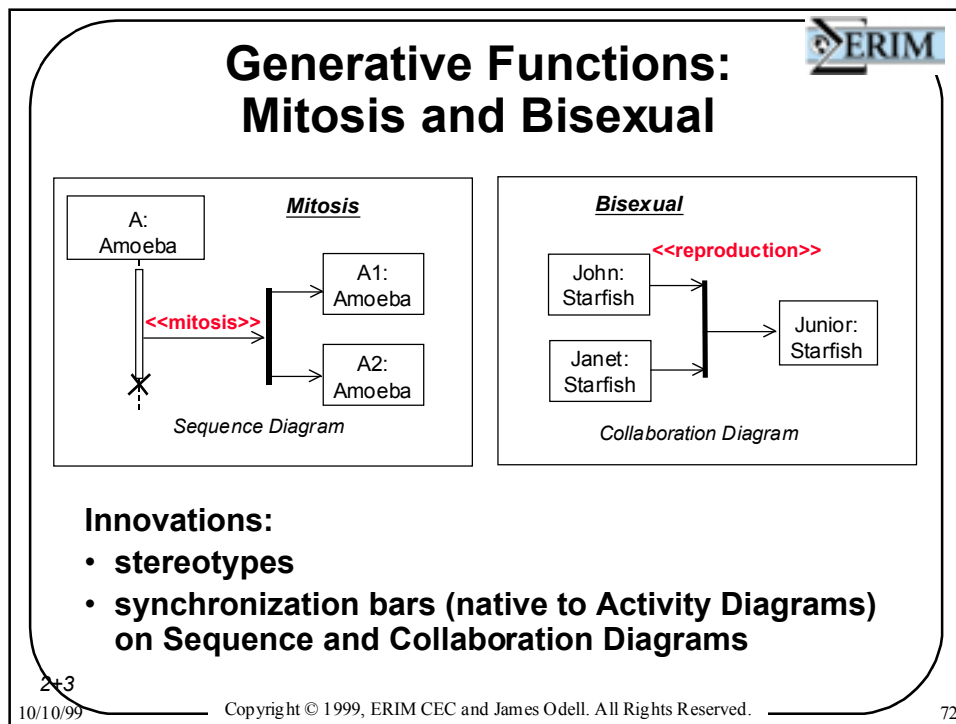
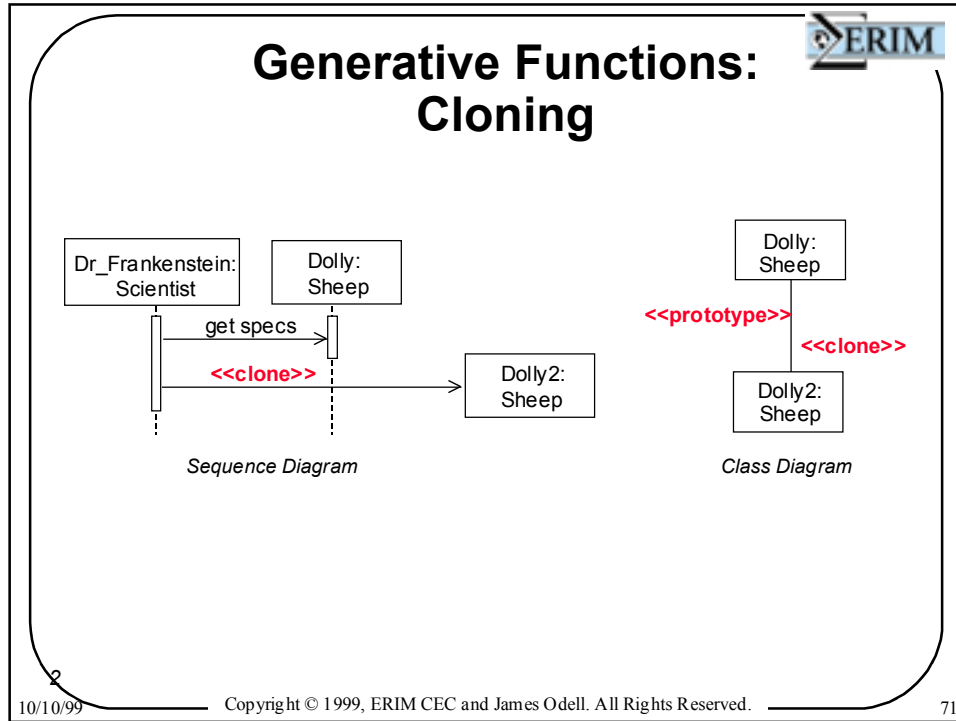


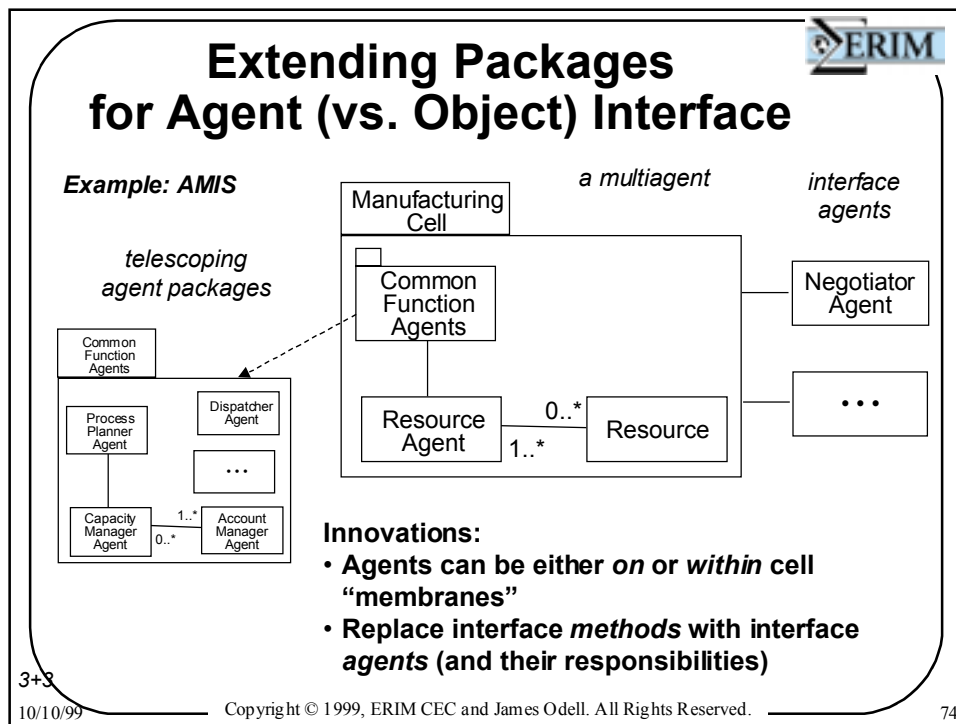
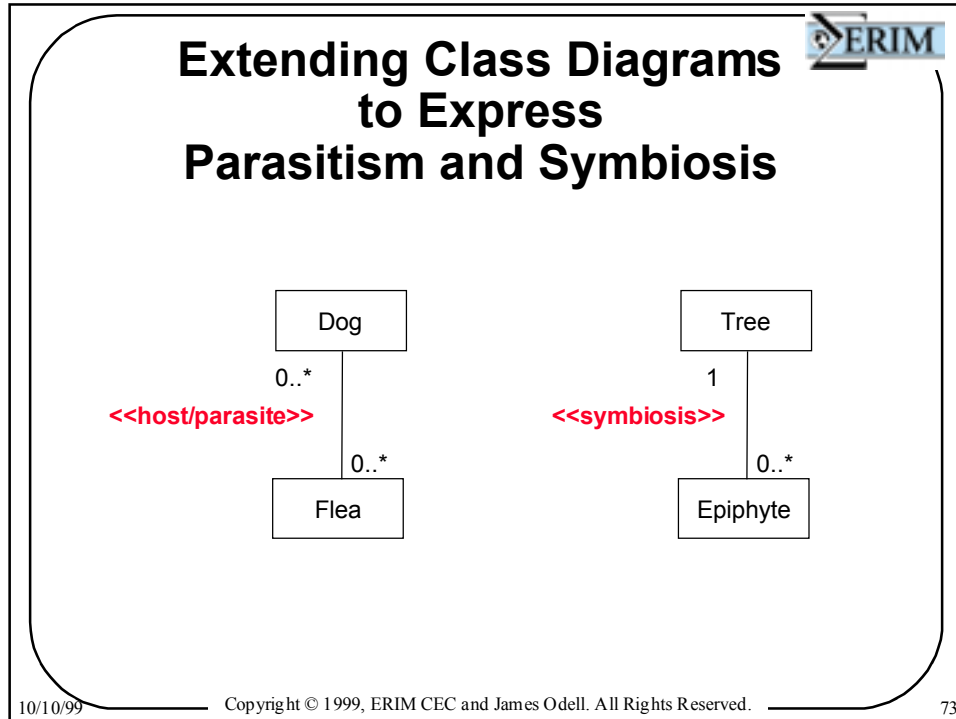












Expressing Emergence by Extending Aggregation in Class Diagrams

```
classDiagram
    Market "0..*" o-- "0..*" Consumer : <<emergence>>
```

10/10/99 Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved. 75

Role Changes

- **Concurrent vs. sequential roles**
 - Concurrent on sequence type
 - Sequential on dashed arrow in collaboration diagram
- **Gets into branching temporal logics; multiple world semantics;**
- **Need either**
 - extension to state or activity diagram
 - new “role diagram”
 - class diagram with dashed line and name of process that causes the change
- **Need examples and list of issues.**

10/10/99 Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved. 76



Generative and Collaborative

- **Currently, use sequence diagram with arrow going to a box. But not joint effort. Could add synch bar from activity diagrams.**

10/10/99

Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

77



Overview

- **Why Engineering Artifacts?**
- **OO Artifacts for MAS Development**
- **Toward Agent-Specific Artifacts**
 - **What do we need to represent?**
 - **What are the most helpful ways to represent them?**
 - **vparunak@erim.org**

10/10/99

Copyright © 1999, ERIM CEC and James Odell. All Rights Reserved.

78