

RE06 Tutorial H3: Building a Platform-Independent Model with MDA

Module 1: Course Overview

Armstrong Process Group, Inc.
www.aprocessgroup.com

Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Course Objectives

2

- Learn fundamental process pattern for object-oriented analysis and design
- Learn how to derive a platform-independent analysis model from use case requirements

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Audience

3

- Anyone directly involved in software development
 - System analyst
 - Use case analyst
 - Project lead
 - Developer/programmer
 - Architect/designer

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Prerequisites

4

- Software development experience
- Knowledge and/or introductory experience with object-oriented concepts
- Understanding of UML and use cases
 - Use Case Modeling Fundamentals (1 day)
 - UML Fundamentals (1 day)
- Continuing education
 - Object-Oriented Design with UML (2 day)
 - User Interface Design with UML (2 day)
 - Test Case Analysis with UML (2 day)

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Building a Platform-Independent Model with MDA

Module 2: Architecture Overview

Armstrong Process Group, Inc.
www.aprocessgroup.com

Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Module Objectives

6

- Define software architecture
- Introduce architecture and design work products
- Introduce 2+9+1 modeling framework
- Introduce Model-Driven Architecture® (MDA®)

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



What is Software Architecture?

- “The software architecture of a program or computing system is the *structure* or structures of the system, which comprise software *components*, the *externally visible properties* of those components, and the *relationships* among them.”

Software Engineering Institute (SEI)
Carnegie-Mellon University

- Refers to abstraction, component, external viewpoint, interacting components, structure, pervasive architecture, and behavior

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved

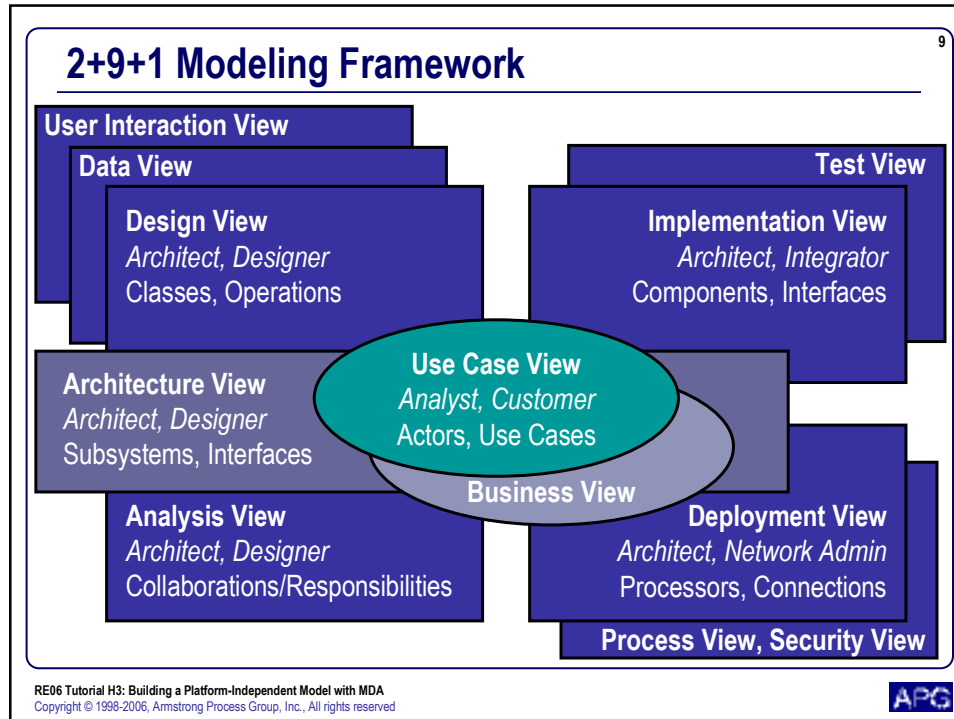


In Short...

- The architecture of a software system can be viewed as a collection of significant components
 - Related and grouped into structures
 - Interact with one another via services that they each provide
 - Aggregate behavior constitutes the behavior of the system
- *Key point: This way of looking at software can be applied at all levels of abstraction and at all degrees of granularity, which is what the remainder of this course focuses on applying*

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved





First Two Views of Architecture

- **Business View**
 - Concerned with external business services and internal workflows
 - Components are business use cases and business object model with workers and entities
- **Use Case View**
 - Concerned with software application behavior and requirements
 - Components are use cases

Next Nine Views

11

- Analysis View
 - Concerned with distribution of use case behavior to preliminary design elements
 - Platform-independent model (PIM)
 - Components are boundary/control/entity classes, collaborations
- Design View
 - Concerned with main logical components and structures in the design of the system
 - Platform-specific model (PSM)
 - Components are classes, operations, packages

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Next Nine Views

12

- Data View
 - Concerned with physical structure of data model
 - Components are databases, tables, relationships, triggers
- User Interaction View
 - Concerned with human factors and interaction
 - Components are windows, web pages, screens, forms

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Next Nine Views

13

- Deployment View
 - Concerned with how the software is distributed across nodes
 - Components are devices, processors, connectors and run-time processes and objects that execute on them
- Process View
 - Concerned with concurrency, real-time response
 - Components are processes, threads

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Next Nine Views

14

- Security View
 - Concerned with authentication/certification
 - Components are roles, privileges, certificates
- Implementation View
 - Concerned with physical implementation
 - Components are UML components (web pages, source, EXE, scripts, command files, tables)
- Test View
 - Concerned with verifying application behavior & structure
 - Components are test cases, test procedures, test scripts

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Architecture View To Integrate All Views

15

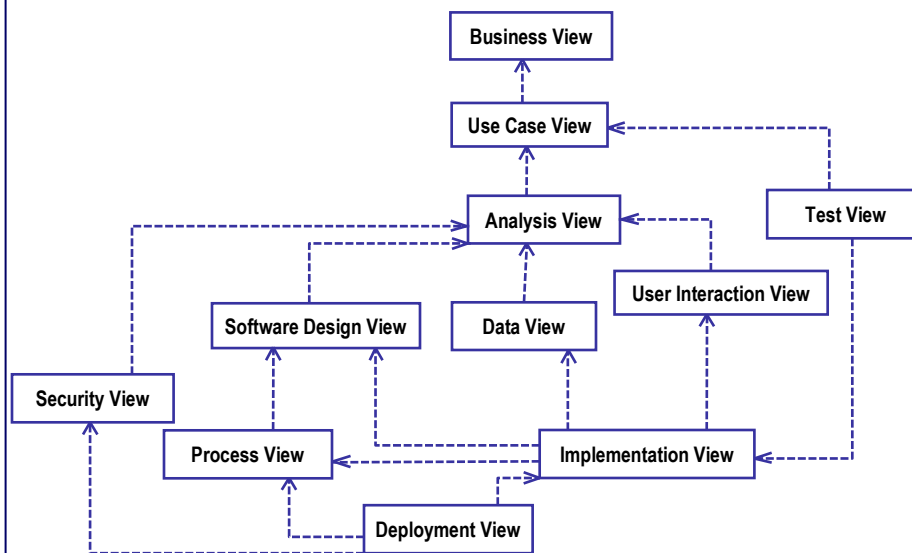
- Separate architecture view
- Shows architectural pattern
- Shows organization and relationships of subsystems and interfaces
- Provides platform-specific transformation rules
- Need to map analysis model to architecture view to generate platform-specific design model

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



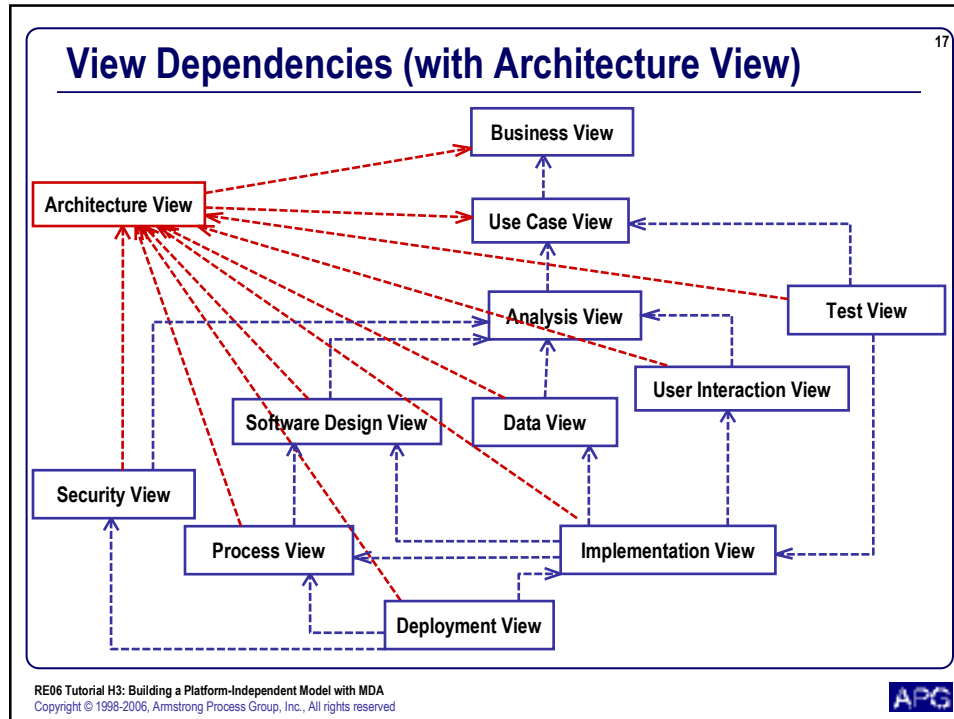
View Dependencies (without Architecture View)

16



RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



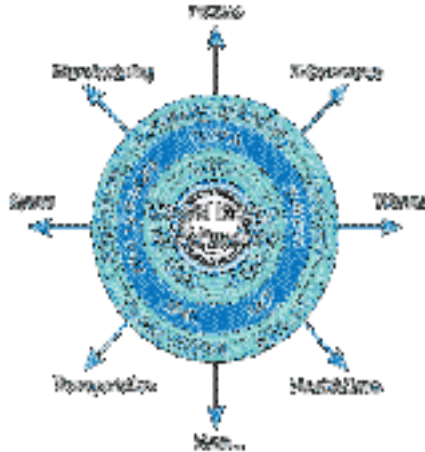


Architecture and Design Work Products

- Analysis
 - Analysis model
 - Analysis realization
 - Analysis class – responsibility
 - Analysis mechanism
- Design
 - Design model
 - Design realization
 - Design class – operation
 - Design mechanism
 - Design subsystem
 - Interface
- Architecture
 - Software architecture view
 - Reference architecture model
 - Architectural pattern realization
 - Subsystem interface realization
 - Subsystem dependency diagram
 - Subsystem usage realization
 - Design pattern realization

Model-Driven Architecture (MDA) Features

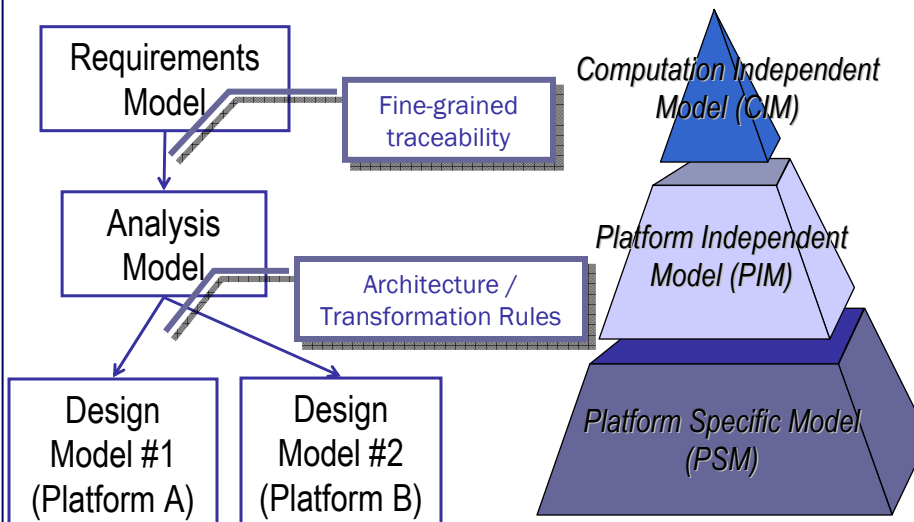
- Ease transition from one platform to another version
- Integrate with existing legacy applications
- Apply domain-specific reference business models
- Use UML 2.0 for all models
- Provide framework for vendor-specific transformation mappings
- Extend useful lifetime of system models
- Supports specialized computing environments



RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved

APG

Project Models And MDA Models



RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved

APG

Building a Platform-Independent Model with MDA

Module 3: Analyze Use Cases

Armstrong Process Group, Inc.
www.aprocessgroup.com

Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Module Objectives

22

- Introduce UML collaborations
- Discuss three types of analysis classes
 - Boundary, control, entity
- Model internal use case behavior
 - Identify analysis class interactions
- Model internal use case structure
 - Identify analysis class responsibilities

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Module Objectives

23

- Identify relationships between analysis classes
 - Associations
 - Aggregation
- Specify association details
 - Navigability
 - Multiplicity
 - Role names

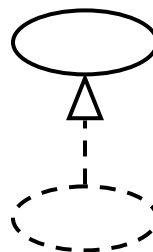
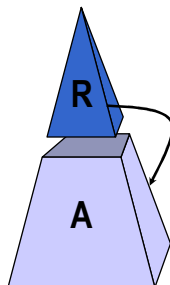
RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Analyze Use Case

24

- Take use cases “down a level”
- How?
 - Apply universal design process pattern!
 - Collaborations are called analysis realizations



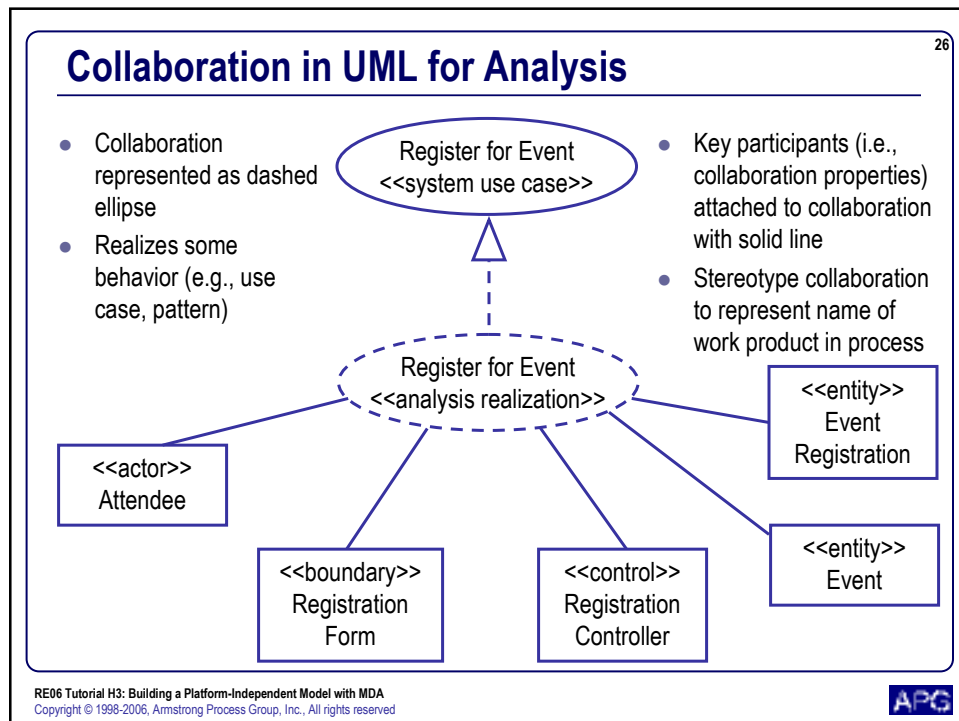
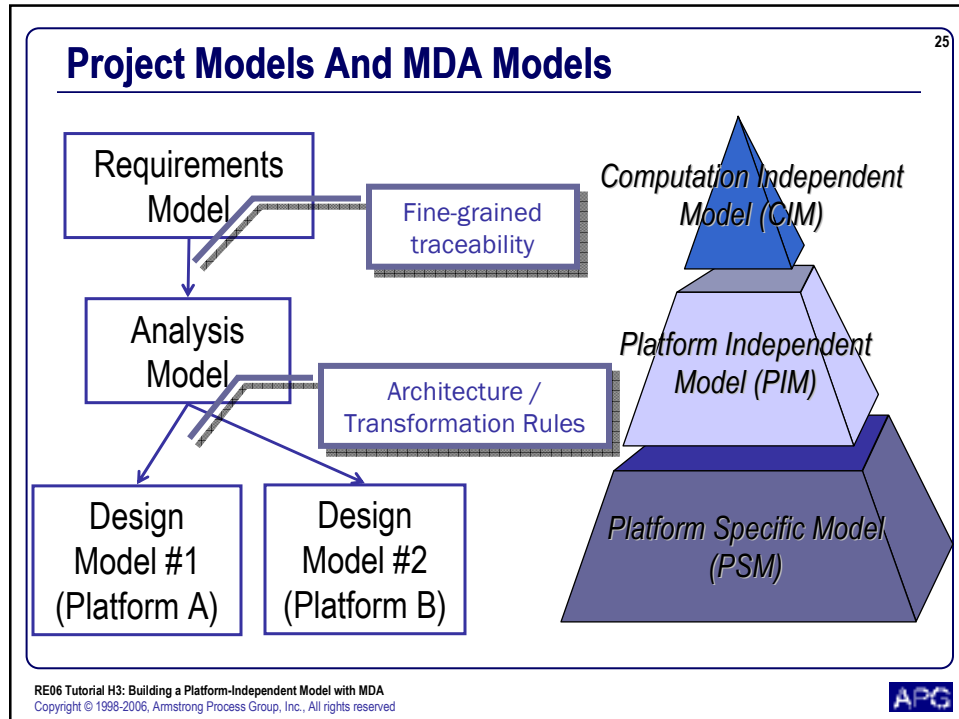
use case

“realizes” relationship

analysis realization

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved





Universal Process Pattern

27

- Given the description of a set of related behaviors
 - Find candidate components
 - Describe their services
 - Describe how they interact
 - Group them into structures to support their interactions
- Such that their aggregate behavior constitutes the set of behaviors in question

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Process Pattern Applied to Use Case Analysis

28

- For each analysis realization
 - Find analysis classes from use case behavior
 - Distribute behavior to analysis classes
- For each resulting analysis class
 - Describe responsibilities
 - Establish associations between analysis classes

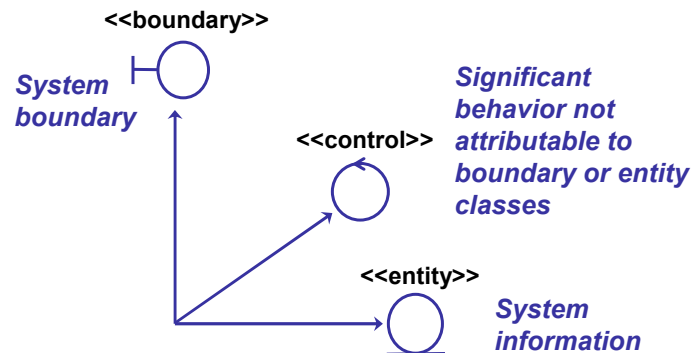
RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



What Is An Analysis Class?

29

- Three orthogonal planes of class behavior



- Jacobson proposed that any class can be characterized strongly along one of these planes

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Find Analysis Classes

30

- Inspect use case diagram and use case specification to determine candidate analysis classes for one use case
- Identify three different types of analysis classes
 - Boundary
 - Control
 - Entity
- Using these three types of classes, completely distribute behavior of use case to analysis classes as responsibilities

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Analysis Classes

31

- Analysis classes are preliminary design classes that have behavior and responsibility
- Focus should be on “what” analysis classes need to do, not “how” they do it
 - The “how” will come in design
- Use three types of analysis classes to help decide where to place certain types of responsibilities

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Boundary Class

32

- Models interaction between things outside system (actors) and system itself
- Start with one boundary class for each actor that communicates with use case
 - Look at use case diagram
- Name boundary classes used by human actors based on what they are being for – typically with a “Form” suffix
- Name boundary classes that communicate with external systems after external system

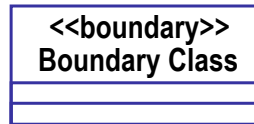
RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



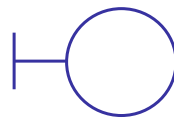
Boundary Class Notation

33

- Class icon with stereotype



- Analysis icon



Boundary Class

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Boundary Class Responsibilities

34

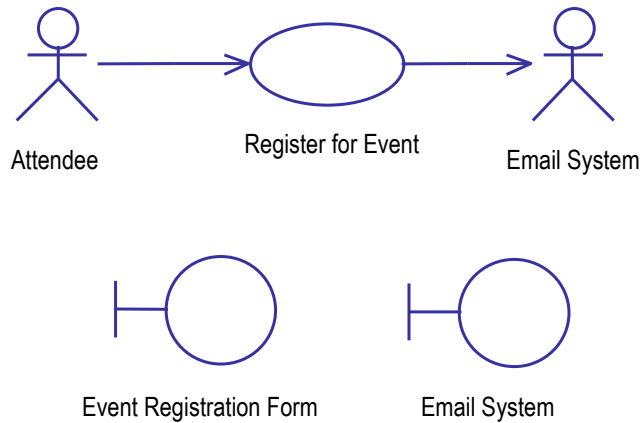
- Boundary classes used by humans should have high-level (abstract) names for their responsibilities
 - Select Thing
 - View Thing
 - Open Thing
 - Enter Thing
 - Display Thing
- Do not use user interface lingo

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Sample Boundary Classes

35



RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Control Class

36

- Controls sequential execution of use case's behavior
- Start with one control class for each use case
- Name control classes after use case with suffix
 - Controller
 - Manager
 - Server

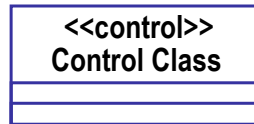
RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



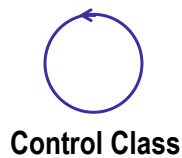
Control Class Notation

37

- Class icon with stereotype



- Analysis icon

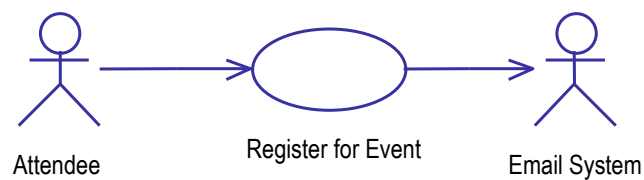


RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Sample Control Class

38



RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Entity Class Responsibilities

39

- Responsible for maintaining state
- Entity classes are usually persistent
- Apply “noun-verb” analysis to use case flow of events
 - Look for active verbs and the nouns they act upon
 - The nouns are candidate entity classes
- Should be at least one (usually more) entity per use case

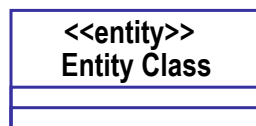
RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



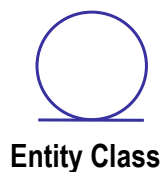
Entity Class Notation

40

- Class icon with stereotype

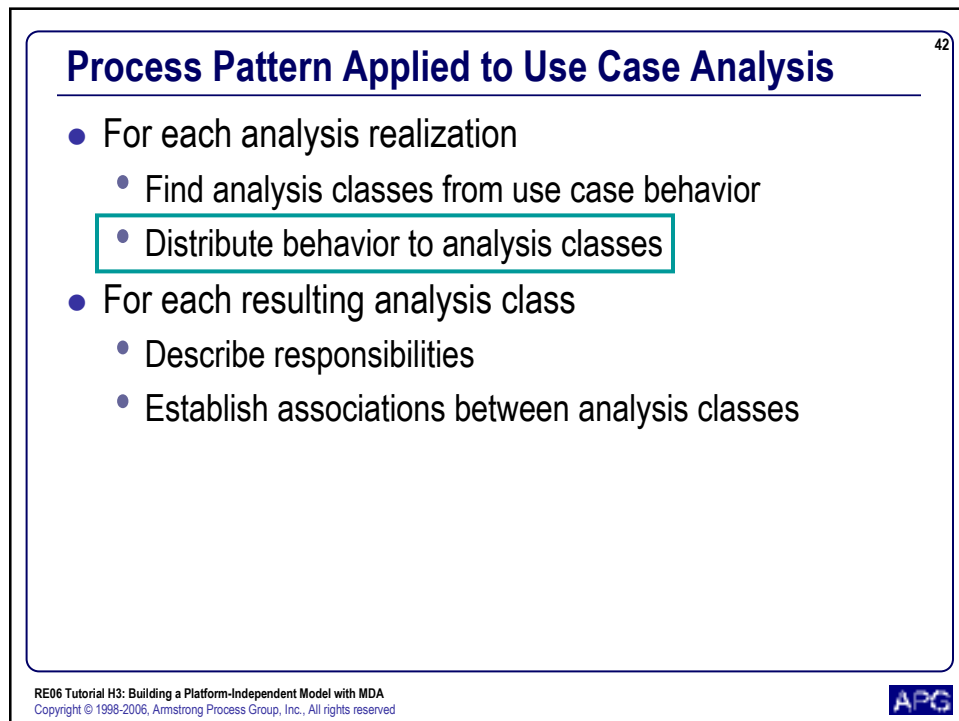
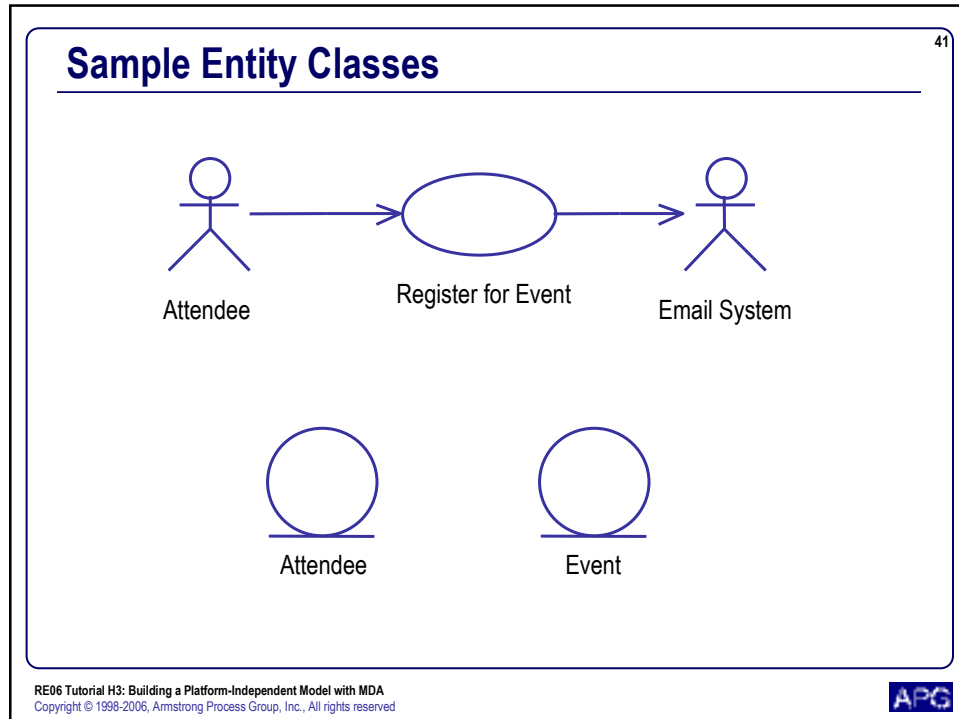


- Analysis icon



RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved





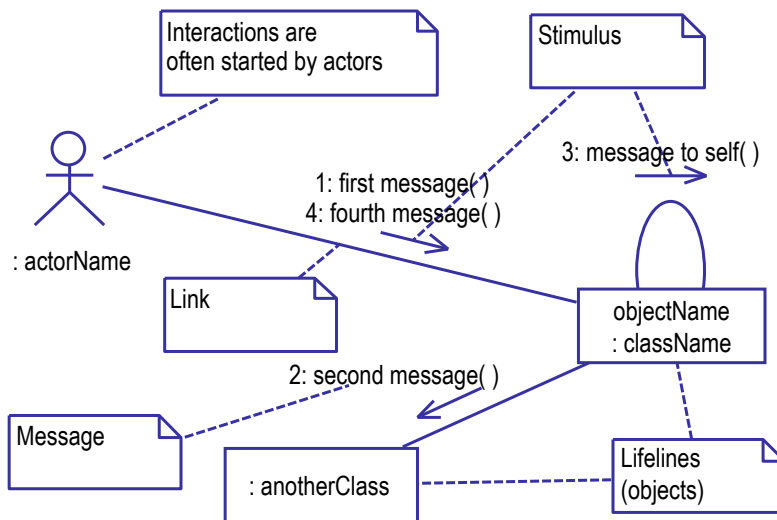
Distribute Behavior

- Express use case behavior in terms of collaborating analysis classes
- Determine responsibilities of analysis classes
- For each flow of events for use case
 - Identify analysis classes
 - Illustrate interactions between analysis classes
 - Create one or more collaboration diagrams

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved

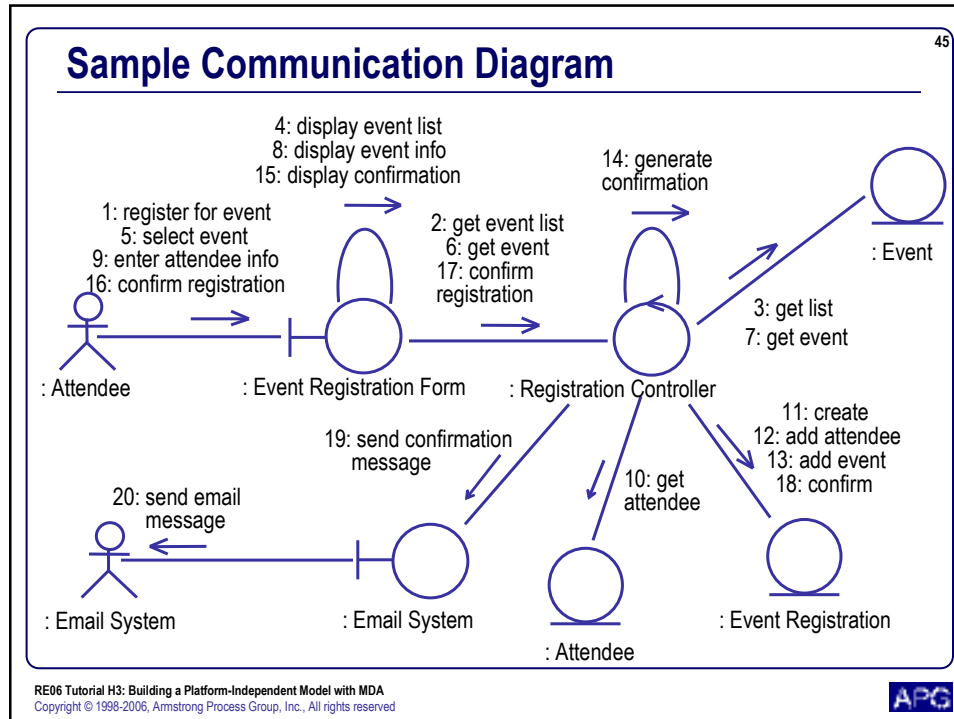


Communication Diagram: UML Notation



RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved





46

Exercise: Model Use Case Behavior

- Given
 - Use case model survey
 - Use case specification (assigned by instructor)
- Identify (for basic flow of use case)
 - Candidate analysis classes
 - Interaction between analysis classes
 - Message sent from one class to another
- Produce
 - List of candidate analysis classes (by type)
 - One collaboration diagram showing interaction

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved

APG

Process Pattern Applied to Use Case Analysis 47

- For each analysis realization
 - Find analysis classes from use case behavior
 - Distribute behavior to analysis classes
- For each resulting analysis class
 - Describe responsibilities
 - Establish associations between analysis classes

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved

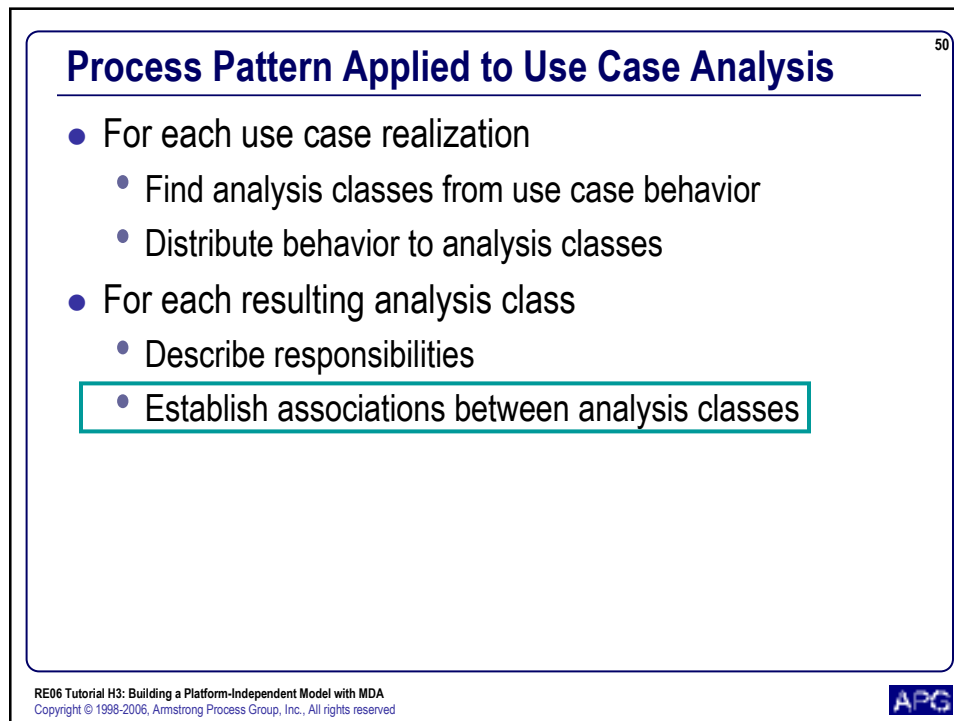
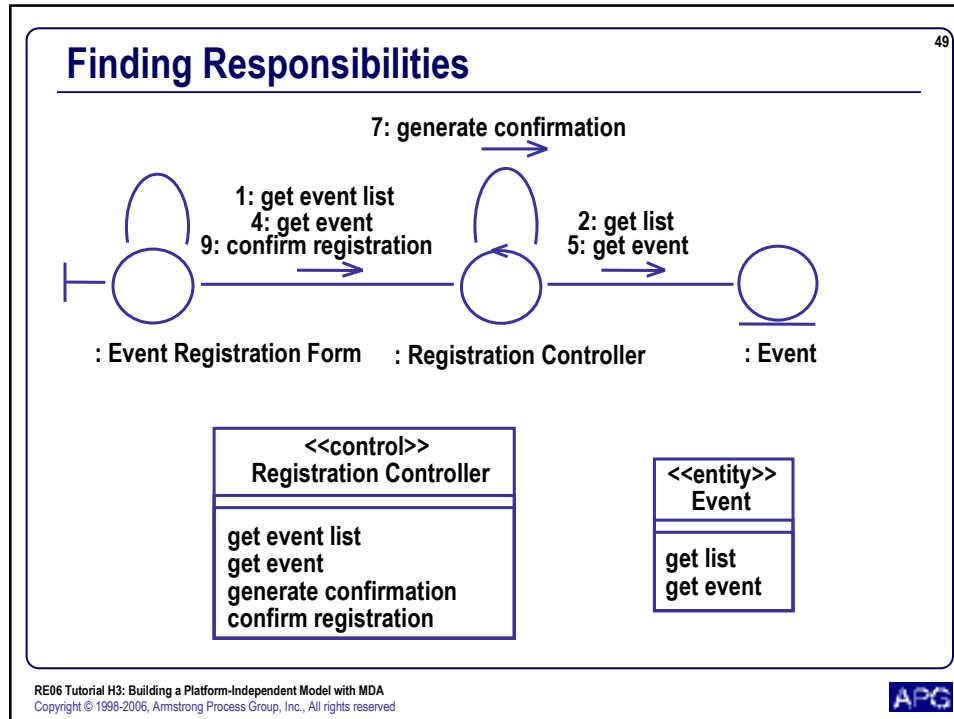


Describing Responsibilities 48

- What are responsibilities?
 - Actions object can perform
 - Commands object can respond to
 - Information object maintains and provides to other objects
 - Abstract description of “what” object does
 - Not “how” it does it – this comes with design and are called operations
- Where do you find responsibilities?
 - Derived from messages in collaboration diagrams
 - Other responsibilities derived from non-functional requirements

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved





Establish Associations

51

- Identify needs for relationships between classes based on links between objects in the collaboration diagrams
 - For one object to send message to another object, relationship must exist between objects' classes
- In analysis, focus on simple relationships
 - Association
 - Aggregation
- Define association characteristics
 - Roles
 - Navigability
 - Multiplicity

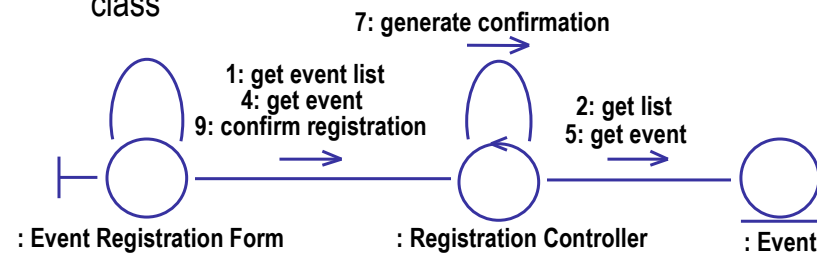
RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Finding Associations

52

- Do not need reflexive association for reflexive link
 - Do need one if link between two objects from the same class



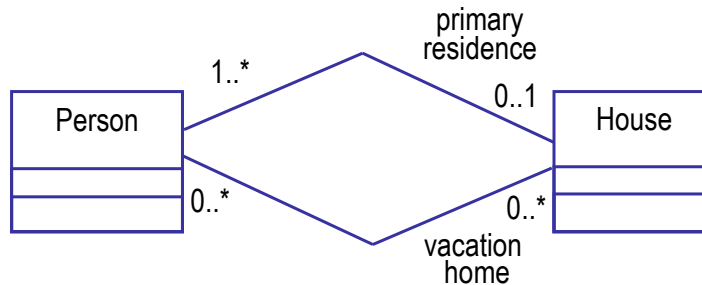
RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Role Names

53

- Used to specify context of the relationship between two objects
- Must use different role names if more than one association exists between the same two classes



RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



Navigability

54

- Indicates that one object can find, or navigate, another object
- Bi-directional
 - Both objects can see each other



An object from Class A can find an object from Class B and vice versa

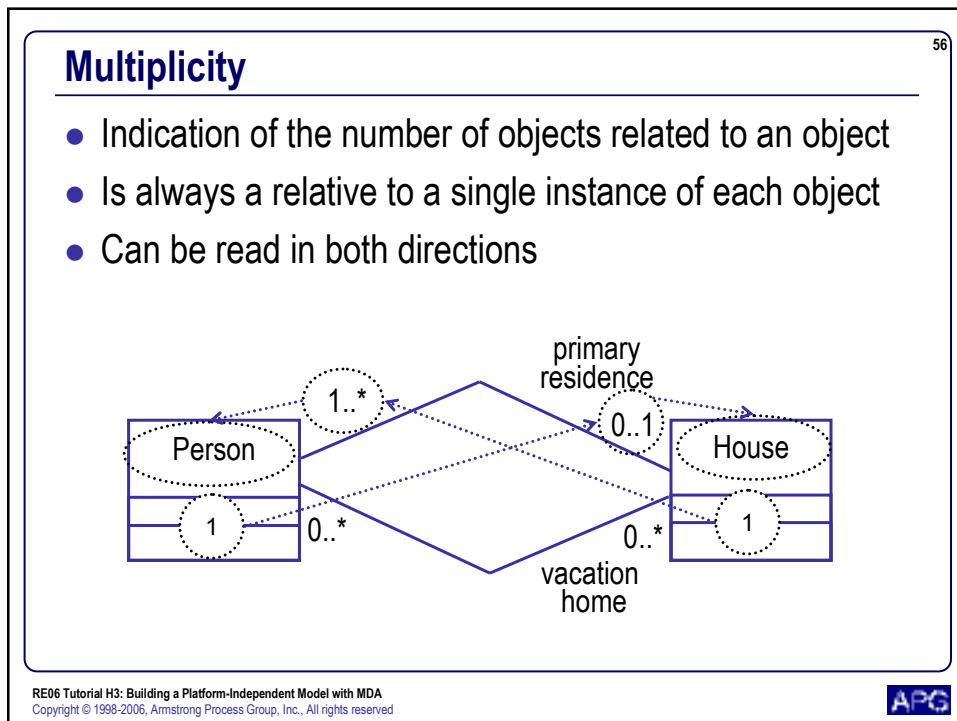
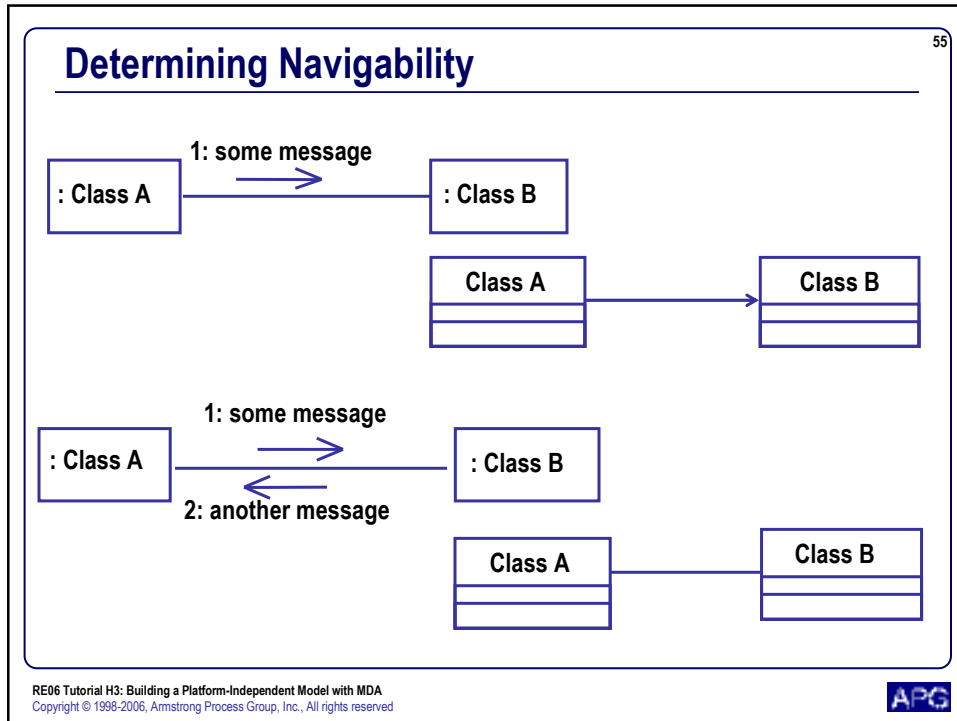
- Uni-directional
 - Only one object can see other



An object from Class A can find an object from Class B, but not B to A

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved

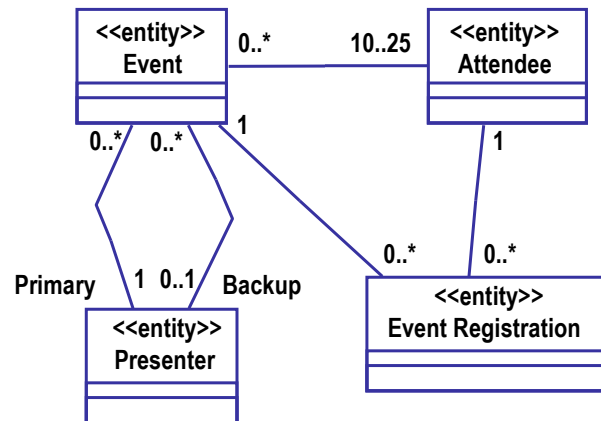




Multiplicity

57

- Indicates how many objects might be related to another object



RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved



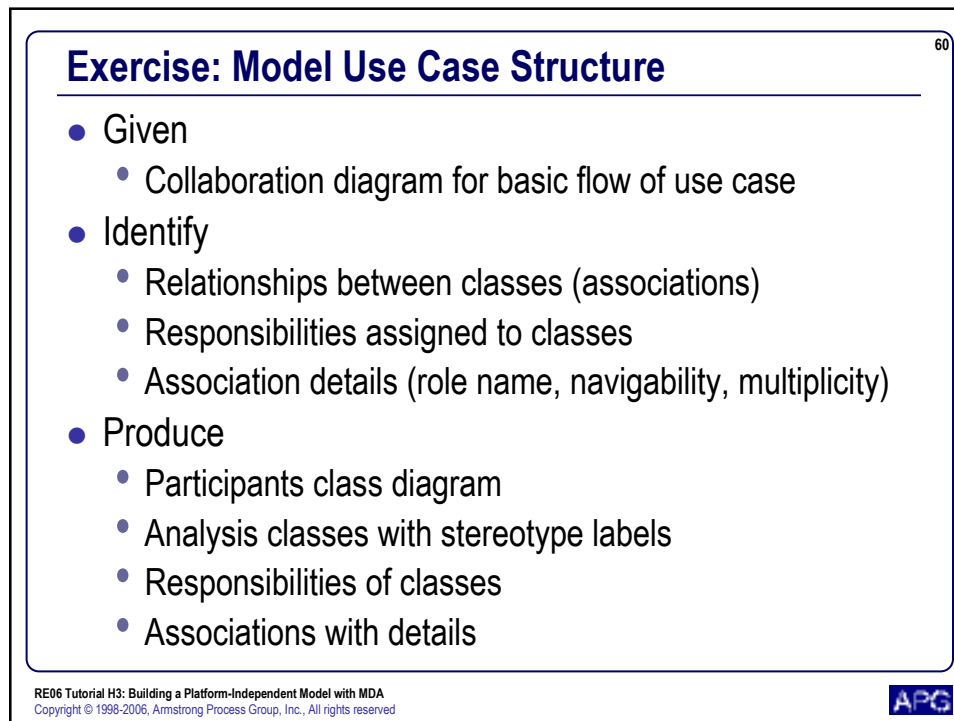
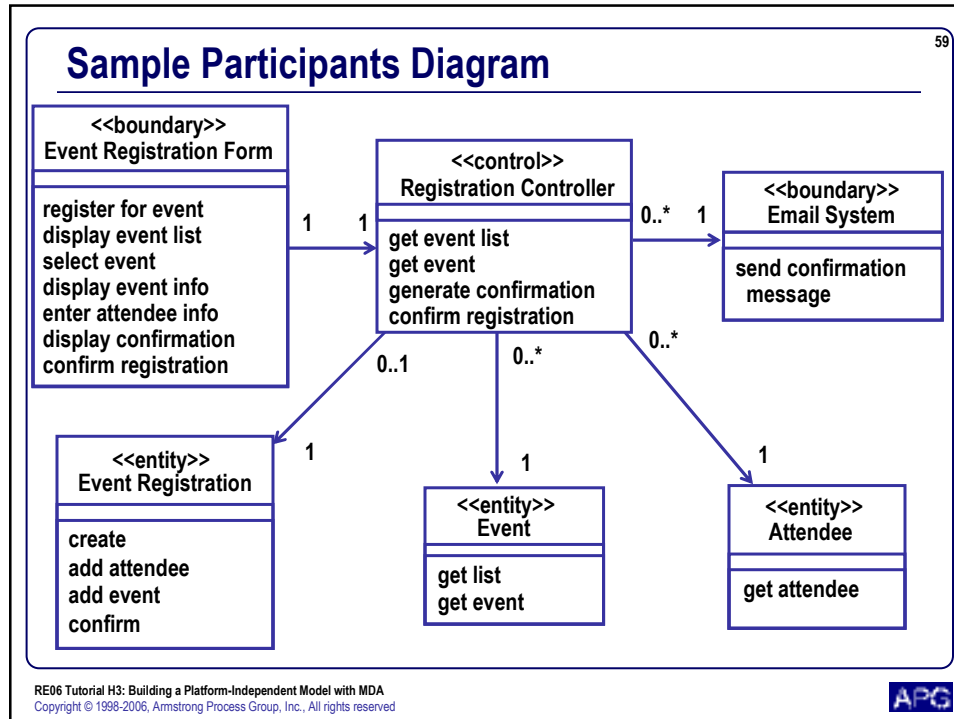
Create Participants Diagram

58

- Each analysis realization must have at least one participants diagram
- Use UML class diagrams
 - Shows structure required to support interactions
- Limit display on participants diagram based on what appears on all interactions diagrams for use case
 - Classes, attributes, responsibilities, and relationships
- Participants diagram is a view into the model
 - Expect same classes to appear on different diagrams for different use cases with different responsibilities and relationships (especially entity classes)

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved





What To Do With Analysis Model

61

- Describe architecture partitioning
 - Identify packages and their contents
- Identify design subsystems and interfaces
- Validate and refine understanding of requirements
- Design user interface
- Verify test cases
- Refine project estimates
- Design data model
- Identify security roles and privileges

RE06 Tutorial H3: Building a Platform-Independent Model with MDA
Copyright © 1998-2006, Armstrong Process Group, Inc., All rights reserved

