User Interface Design with UML

Fourth Workshop On UML for Enterprise Applications

ATC Enterprises, Inc.
7402 Borman Avenue
St. Paul, MN 55076
651.554.1771
www.atcenterprises.com

Objectives

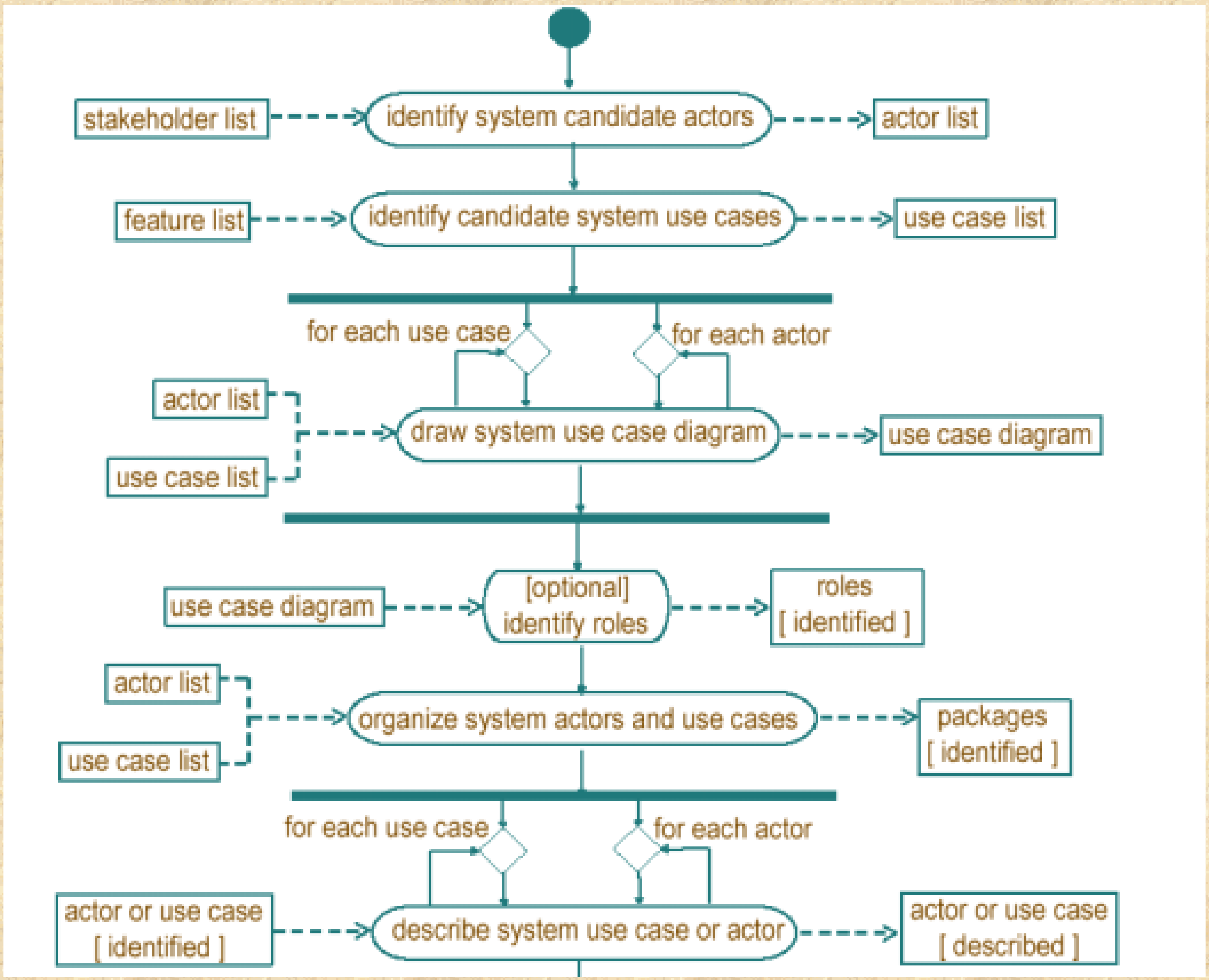
- Process overview and
- Tips for detailing use cases and how to handle user interface (UI) requirements
- Using UML collaborations for UI design
  - Identify UI elements
  - Identify their responsibilities
  - Identify their relationships
- Using Rational Rose, RequisitePro, and SoDA

Adaptive Team Collaborative Process^{SM} (ATCP^{TM})

- Practical, agile, and iterative
- Use case based and risk driven
- Customer-centric and adaptive
- Applies practical UML guidance
- Supports MDA through traceability strategy and tool usage
- Based on Software Process Engineering Metamodel (SPEM)
- Will be available open license/freeware
- Leverages tool transparency for real-time measurement

- Extreme Programming (XP)
- Adaptive Software Development (ASD)
- Usage-Centered Design (UCD)
- Object-Oriented Analysis and Design (OOAD)
- Unified Modeling Language (UML)
- Rational Unified Process (RUP)
- Capability Maturity Model (CMM)
- UCD adopted ATCP Actor/Role UML notation – Oct 2002
  - http://www.foruse.com/newsletter/foruse26.htm

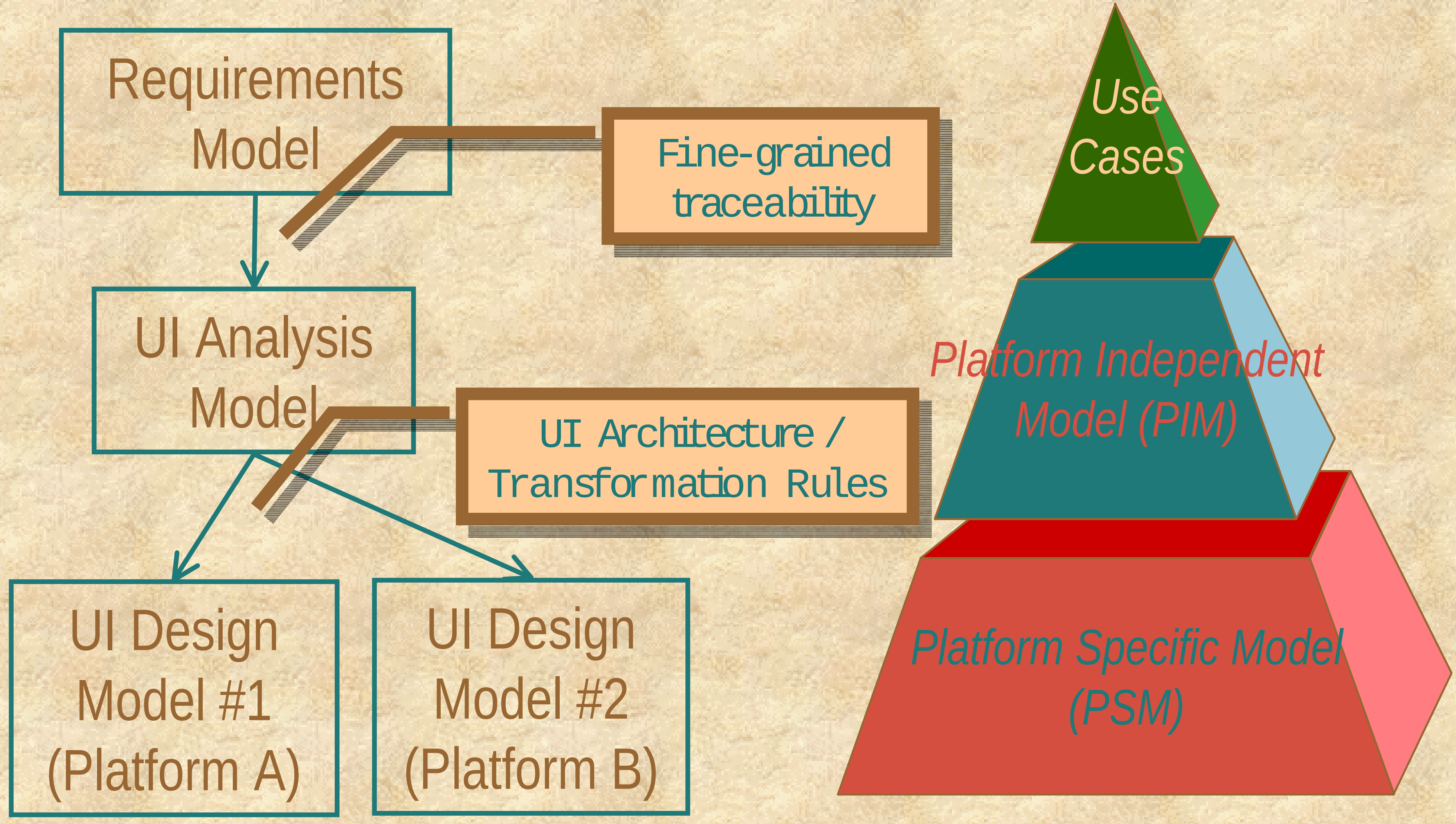
Sample ATCP Workflow


stakeholder list
identify system candidate actors
actor list
feature list
identify candidate system use cases
use case list
for each use case
for each actor
actor list
use case list
draw system use case diagram
use case diagram
use case diagram
[optional] identify roles
roles [identified]
actor list
use case list
organize system actors and use cases
packages [identified]
for each use case
for each actor
actor or use case [identified]
describe system use case or actor
actor or use case [described]

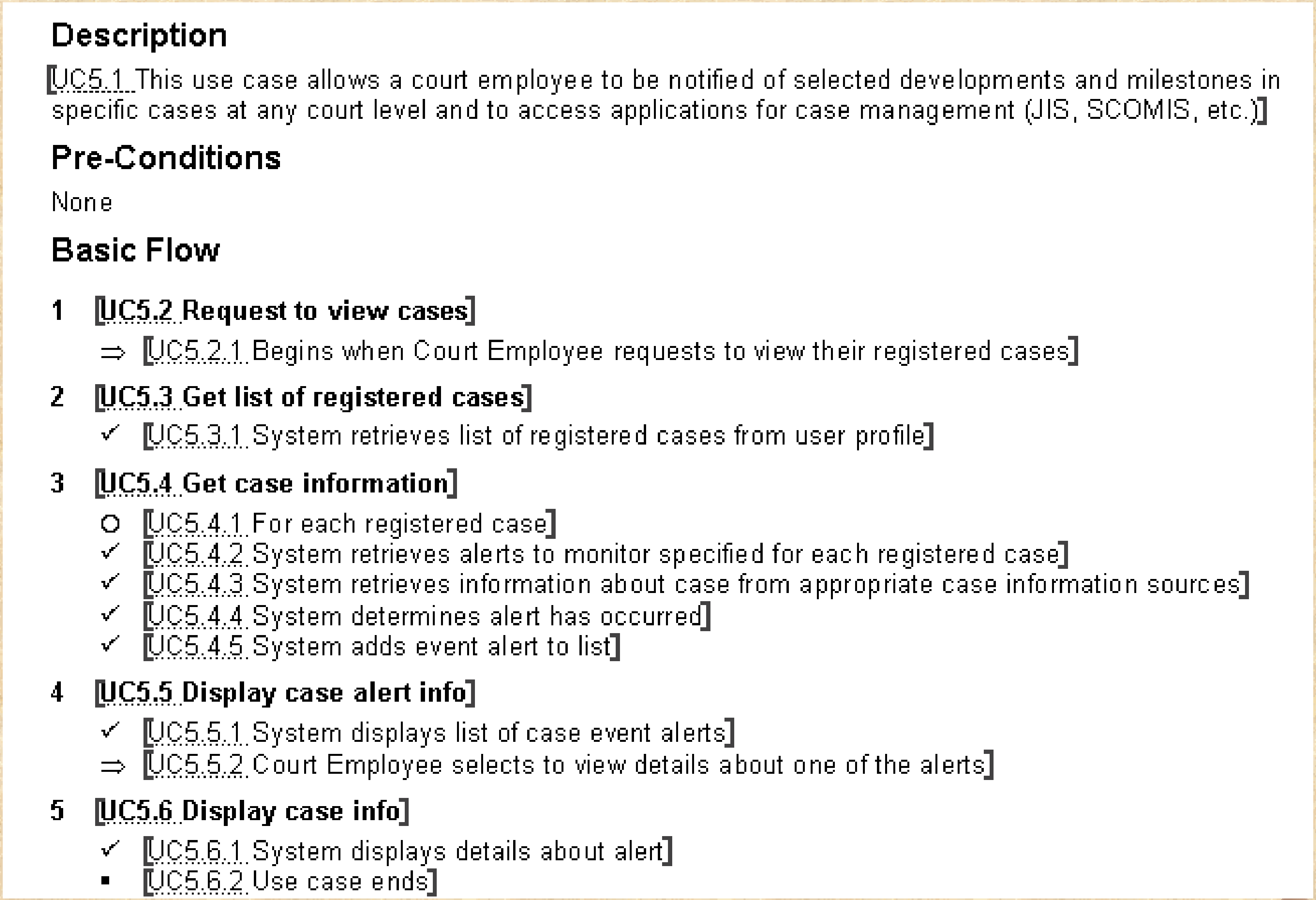
User Interface Analysis and Design

- ATCP recognizes user interface analysis and design as separate discipline
- Follow very similar process pattern as in object-oriented analysis and design
  - However, do not focus on what happens “inside” system – just at the system boundary
- Similar to steps 6-8 in Agile UCD process
- Build the content model and navigation map using UML

ATCP User Interface And MDA


Requirements Model
Fine-grained traceability
UI Analysis Model
UI Architecture / Transformation Rules
UI Design Model #1 (Platform A)
UI Design Model #2 (Platform B)
Use Cases
Platform Independent Model (PIM)
Platform Specific Model (PSM)

ATCP Use Case Specification



Description

[UC5.1 This use case allows a court employee to be notified of selected developments and milestones in specific cases at any court level and to access applications for case management (JIS, SCOMIS, etc.)]

Pre-Conditions

None

Basic Flow

1 **[UC5.2 Request to view cases]**

⇒ [UC5.2.1 Begins when Court Employee requests to view their registered cases]

2 **[UC5.3 Get list of registered cases]**

✓ [UC5.3.1 System retrieves list of registered cases from user profile]

3 **[UC5.4 Get case information]**

○ [UC5.4.1 For each registered case]
✓ [UC5.4.2 System retrieves alerts to monitor specified for each registered case]
✓ [UC5.4.3 System retrieves information about case from appropriate case information sources]
✓ [UC5.4.4 System determines alert has occurred]
✓ [UC5.4.5 System adds event alert to list]

4 **[UC5.5 Display case alert info]**

✓ [UC5.5.1 System displays list of case event alerts]
⇒ [UC5.5.2 Court Employee selects to view details about one of the alerts]

5 **[UC5.6 Display case info]**

✓ [UC5.6.1 System displays details about alert]
▪ [UC5.6.2 Use case ends]

ATCP Use Case Steps

- ⇨ Actor-initiated step (user intention)
- ✓ System response
- ✦ Alternate condition
- ▪ Next step
- ○ Loop/iterate

User Interface Analysis

- Create user interface realization
- Identify candidate UI elements
- Model role-boundary interaction
- Re-factor UI responsibilities
- Model UI navigation

Architecture Process Pattern

Given the description of a set of related behaviors

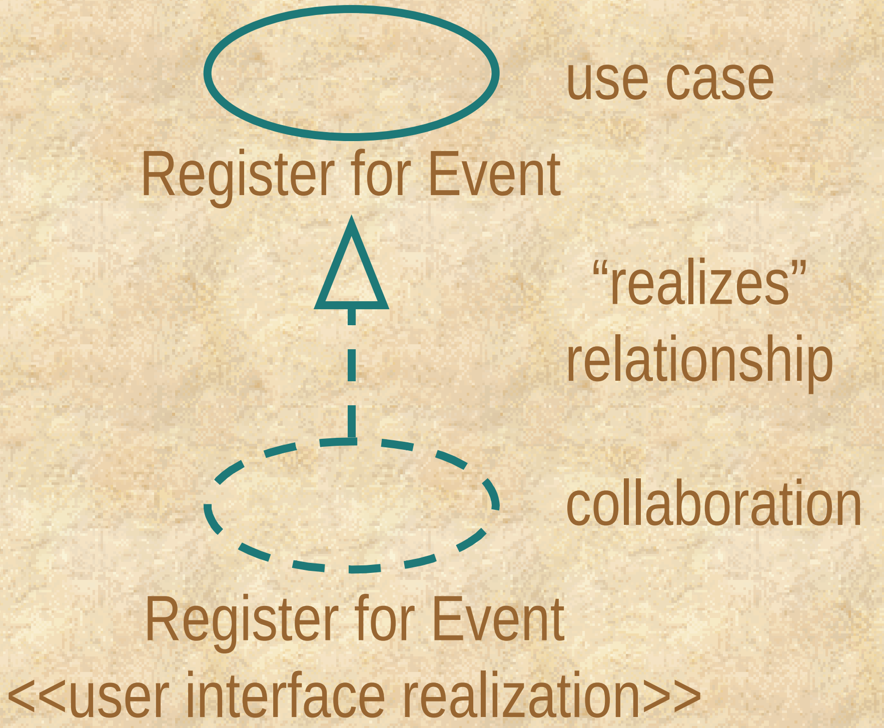
- Find candidate components
- Describe their services
- Describe how they interact
- Group them into structures to support their interactions

such that their aggregate behavior constitutes the set of behaviors in question

This way of looking at software can be applied at all levels of abstraction and at all degrees of granularity; ATCP applies this viewpoint to derive one model from another

Create User Interface Realization

- Use UML collaboration to capture the behavior and structure of the user interface model
- Stereotype the collaboration as <<user interface realization>>
- Connect to use case using realization relationship

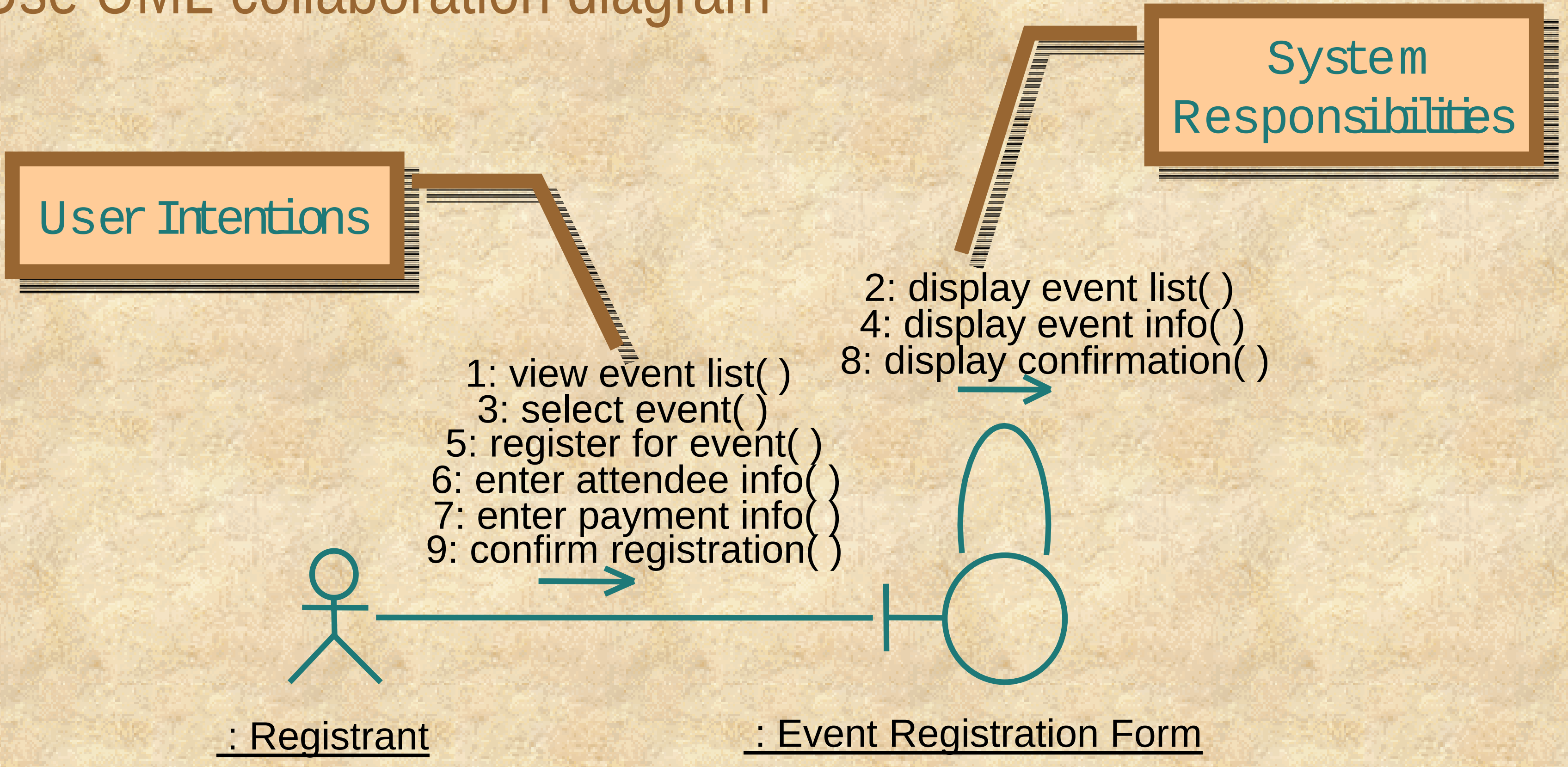


UI Analysis Model Stereotypes

- View
- Form
- List
- Menu

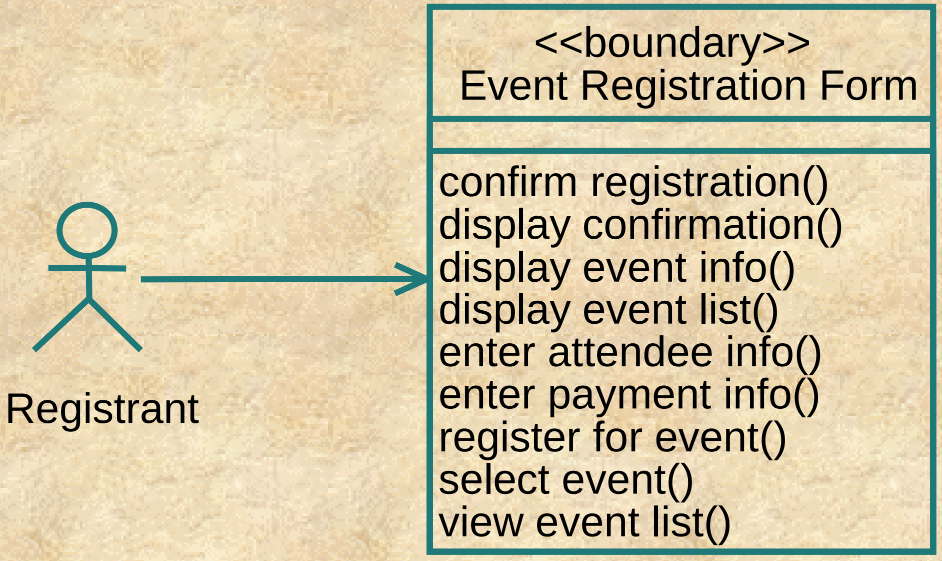
Model Role-Boundary Interaction

- Use UML collaboration diagram



Assign User Interface Responsibilities

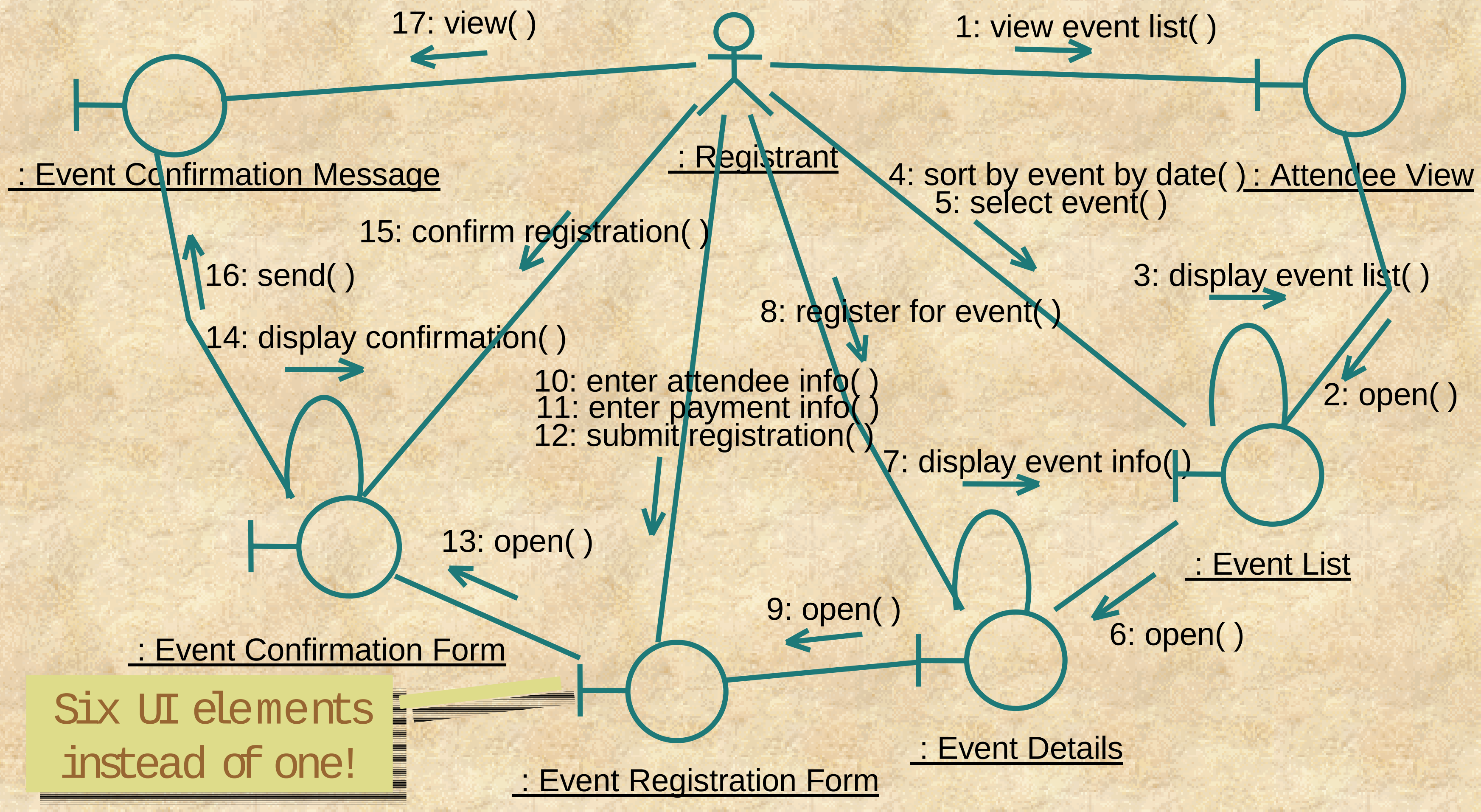
- Use UML class diagram
- Look messages that are going to objects
  - They become responsibilities of the destination object
- Model as operations in a class icon



Re-Factor User Interface Responsibilities

- We followed a pretty simple guideline to get started
  - One boundary class per use case per actor
- Following this guidelines helps us focus on identifying responsibilities
- Inspect them to see if they could be re-factored to a larger number of more fine-grained user interface elements
- Support engineering principles
  - Loosely coupled
  - Highly cohesive

Re-Factor User Interface Responsibilities

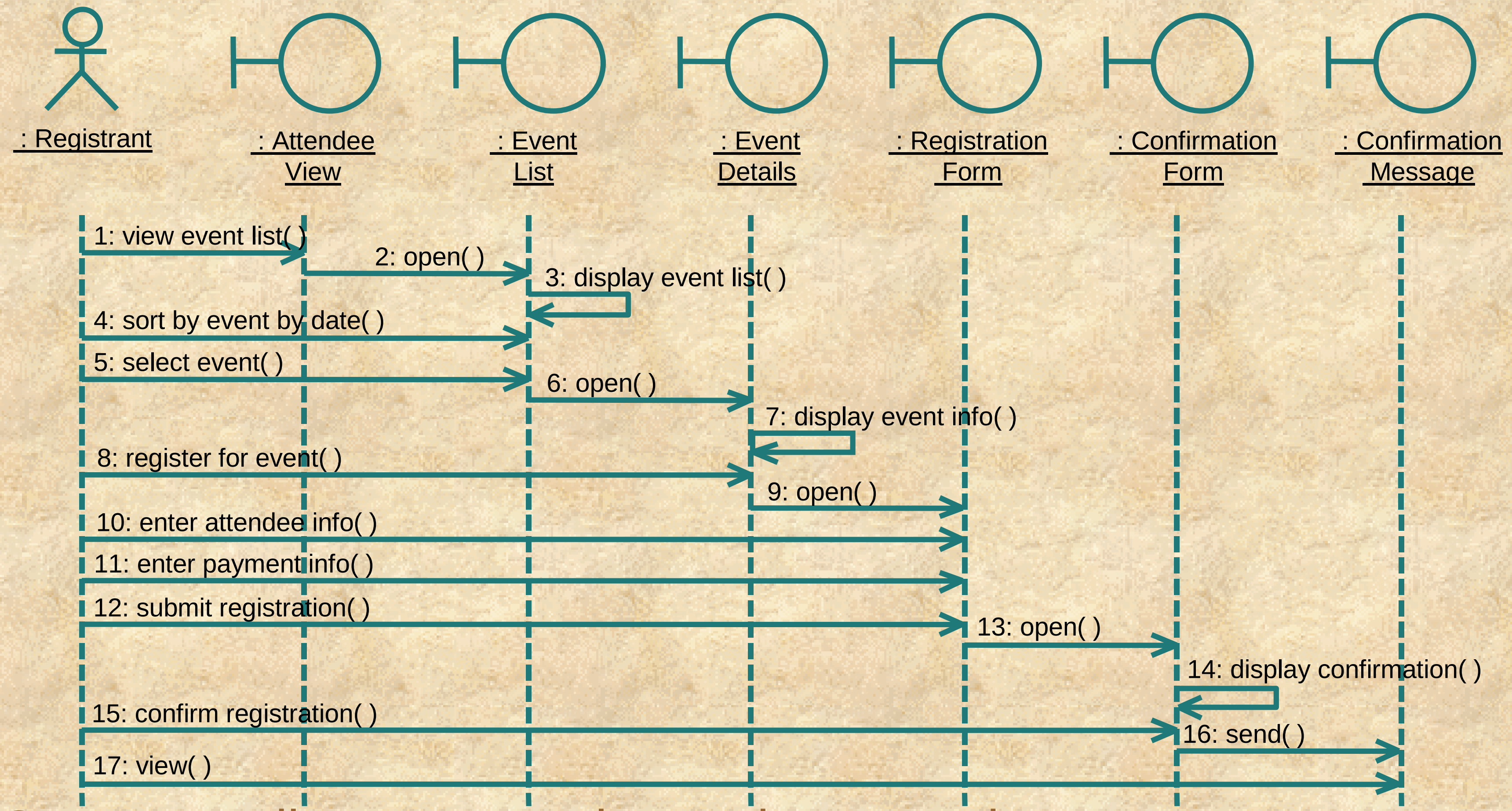


17: view( )
1: view event list( )
: Registrant
: Event Confirmation Message
4: sort by event by date( )
5: select event( )
: Attendee View
15: confirm registration( )
16: send( )
3: display event list( )
8: register for event( )
14: display confirmation( )
10: enter attendee info( )
11: enter payment info( )
12: submit registration( )
2: open( )
7: display event info( )
13: open( )
: Event List
9: open( )
6: open( )
: Event Confirmation Form
: Event Details
: Event Registration Form

Six UI elements instead of one!

- Collaboration diagrams show patterns of interaction
  - Shows relationships in addition to behavior
  - Easy to draw free hand

Re-Factor User Interface Responsibilities

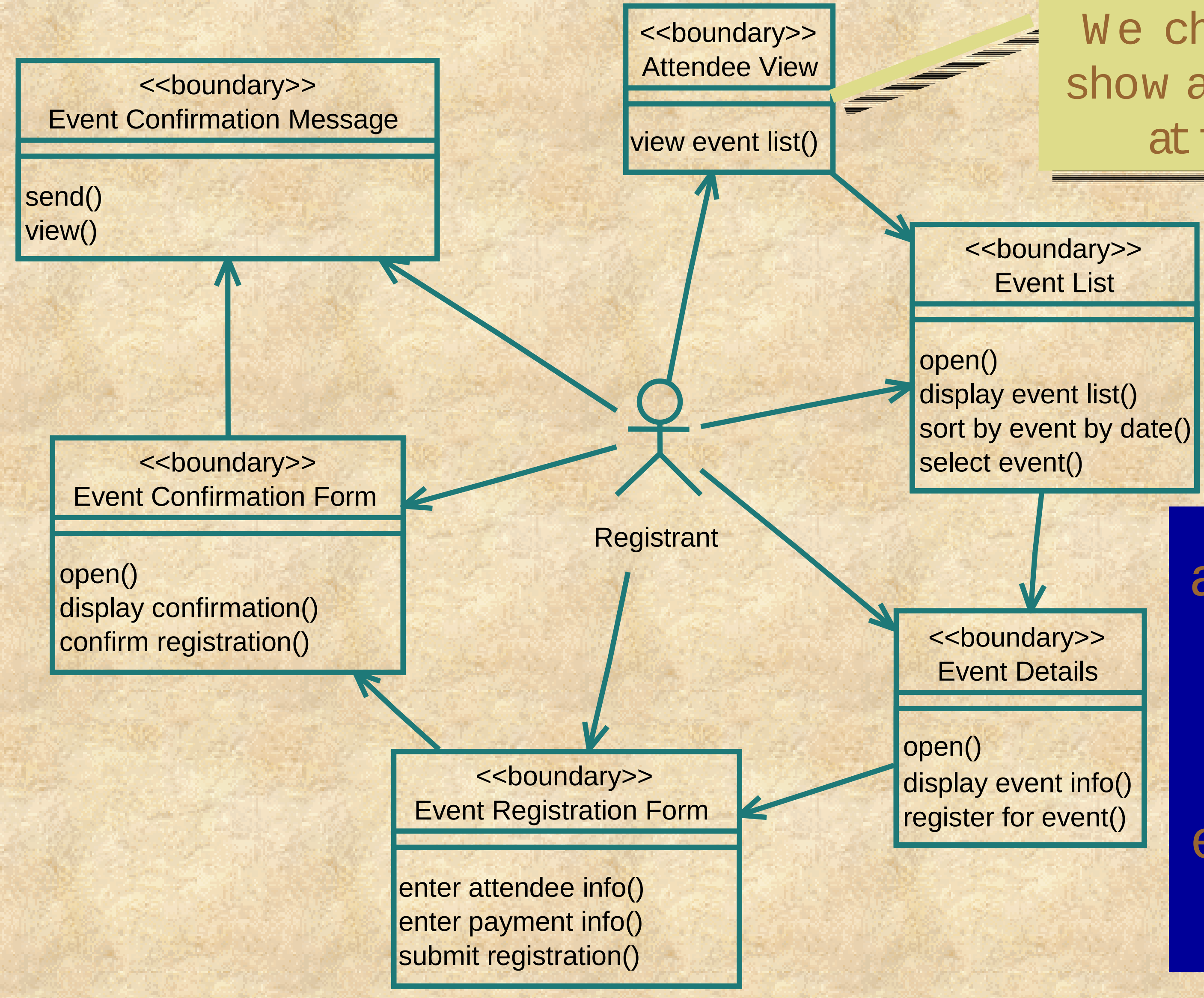


- Sequence diagrams can be easier to read
  - Do not show relationships
  - Not as easy to draw free hand

Model User Interface Navigation

- After creating several user interface realizations for a set of use cases
  - Look for common responsibilities and continue to re-factor
  - Use generalization and composition
- Can create a navigation map that spans multiple use cases for overall flow
  - Use UML class diagram

Model User Interface Navigation



We choose not to show any attributes at this time

Use attributes to show which data elements to display

User Interface Design

- Platform-specific model
- Influenced by UI architecture, standards and conventions, and platform constraints
- Many possible design models for one analysis model
- Apply similar technique as in analysis
    - Use collaborations!
- Use new stereotypes for classes
    - Dependent on platform
    - <<web page>>, <<ASP>>, <<JSP>>, <<window>>
    - <<select>>, <<menu>>, <<tab>>, <<dialog>>, <<treeview>>

User Interface Architecture

- UI architecture patterns
- UI architecture mechanisms
- UI design patterns
- Layout/style/graphic conventions
- Controls transformation of PIM to PSM

UI Architecture Patterns

- Business architecture
  - Knowledge portal
  - Customer service
  - Retail sales
  - Marketing
  - Communities
  - Auction
  - Search
  - Workflow
- Software architecture
  - Model-View-Controller (MVC)
  - Layered / n-tiered
  - Broker
  - Reflection

UI Architectural Mechanisms

- Display object
- Window management
- Dialogue support
  - Keyboard
  - Voice recognition
  - Handwriting recognition
  - External devices
- Security
- Error handling
- User support
  - Computer-based training
  - On-line help
  - Interactive customer service
- Printing
- Personalization
- Session management
- Quality of service (QoS)
- Navigation

User Interaction Design Patterns

- Hypertext
- Kiosk
- Window
  - Multiple document
  - Single document
- Interactive voice response (IVR)
- Batch
- Mobile
- Real-time
- Personal digital assistant (PDA)

MDA Tool Support

- Automatically create UI analysis model structure from use case flows in requirements management tool
  - Create user interface realization and traceability diagram
  - Create individual sequence diagrams for each flow
  - Create single class diagram
- Establish traceability from flow in RM tool to sequence diagram in modeling tool
- Support instant generation of UI design reports
- Capture UI design patterns and architectural mechanisms
- Automate transformation from analysis to design
- Generate functional UI from design model

Questions?

Thank you for your attention and participation!