Lecture 4: Sequence Interaction
Diagrams
Behavioral vs Static Models

- Describe interactions between entities
- Sequence Diagrams
- Collaboration Diagrams
- Message Sequence Charts
- State Charts

- Describe structure of complex entities
- Class Diagrams
- Entity-relationship models
Sequence Diagrams - Components

• Objects
  – Actors, subsystems, class instances

• Messages
  – One object sends a message to another message

• Each diagram describes a scenario, or what happens during a use of the system

• System described by a collection of diagrams
Set member data scenario

Preceded by and followed by LogOn and LogOff scenarios

setMemberData()

setMemberData(dateeData, name)

getMemberData(name)

updateMemberData(memberData)
Applicability

• System/Actor interaction
  – Requirements elaboration

• Subsystem interactions
  – Requirements elaboration and design

• Subsystem design
  – Object properties and class visibility
System Actor Interaction

- Investigate the interactions the user has with the system
- Specify the needed user interface forms
  - Specify contents/capabilities of GUI before graphical design
- Following example note: In the actor/system interaction we see self-messages notation
Enter member data scenario

- Enter Member Data form
  - name, dateData
  - result = enterMemberData

- Display result message (see comm)
Subsystems Interaction

- Show the messages that the subsystems send back and forth
- Enforce a layers architecture
- Identify the subsystems interface/proxy
- In the following example we see the alternatives notation, which allows multiple related flows to be included in
Additional Examples

• Get a Date Scenario
  – Includes conditionals allowing alternative paths in same scenario/ISD
• Start up Scenario
  – Shows interactions with system
• Log On Scenario
• Exit/Log Off Scenario
Start up scenario

At this point follow LogOn/LogOff scenarios
Log Off scenario
Additional Notation

• **Focus of Control**
  – Indicate when an object’s lifestream is “active”

• **Data return from a sent message**

• **Iteration of message**
Systematic Object Identification for Sequence Diagrams

• Rational Software approach (Rational now part of IBM and source of Rational Unified Process)

• Four kinds of objects
  – Actor
  – Boundary/Form
  – Entity/Data
  – Control
Boundary Objects

• Place where the actors interface with the system
• Referred to as forms by some methods
Entity

- Data type objects
- Place where information is stored
- Data Bases
Control

• Coordinates the activities of the use case
• Often identified with a use case, with one per use case
• Possible controller objects in O/O
  – Use Case, Role, System/subsystem, and Business
Additional Notations – Message Synchronization

- Synchronous: sender waits until receiver has processed message
- Procedure call: receiver is ready to receive call, sender waits for return from call. A kind of synchronous
- Asynchronous: sender sends message and then continues regardless of status of receiver
- Balking: if receiver is not ready to receive message it is abandoned
Application to Distributed Systems

- Sequence diagrams show collections of interactions between components in system, like subsystem interaction diagrams
- Can use to develop specifications for components, with individual state diagrams
DS Scenarios For Distributed System

• Scenario 1
  – User who is a member logs on and asks for a date. A date is found. Client relies on server to supply first and subsequent records from member Data base.

• Scenario 2
  – User logs on who is not a member. Then a user who is a member logs on. The client eventually asks the server for the next record and nil is returned since there are no more. Client displays a sorry-no-date message.
Developing a State Model Specification for a Component

• Factor the sequence diagrams, e.g. for the Client, get all the client lifelines

• Decide where the states are
  – E.g. For Client, each new GUI Frame?
    • Not always an exact match with the traces
  – After each message received
Client State Diagram from Traces 1 and 2: