Software Engineering with Objects and Components

Massimo Felici

Room 1402, JCMB, KB

0131 650 5899

mfelici@inf.ed.ac.uk

Administration

- Look at the SEOC1 course webpage:
  http://www.inf.ed.ac.uk/teaching/courses/seoc1.html
- **Tutorials** start next week; Frequency: once a week
- Course Resources:
  - **Main Course Book**: UML, Schaum's Outline Series, Simon Bennett, John Skelton and Ken Lunn, McGraw-Hill, 2001, ISBN 0-07-709673-8.
  - Course Book, Lecture Notes and References
- **Assessment**:
  - 25% coursework; 75% degree examination
- **Coursework**: in small teams (approx 3-5 people); two deliverables equally weighted
  - 1st deliverable: **Monday, 25th October**
  - 2nd deliverable: **Monday, 6th December**
- **Software**: Argo/UML and Java

Software Engineering

- Software Engineering Institute motto:
  - The right software. Delivered defect free, on time and on cost, every time.
- Software Engineering studies:
  - How to make software that is fit for purpose.
  - "fit for purpose": good enough – functionally, non-functionally, meets constraints of the environment, law, ethics and work practice.
  - How to meet time and financial constraints on delivery.
- We still fail too often
  - see a Collections of Software Bugs by Prof. Thomas Hackle

[web link from the SEOC1 webpage]

An Example: Patriot Missile

- On February 25, 1991, during the Gulf War, an American Patriot Missile battery in Dharan, Saudi Arabia, failed to track and intercept an incoming Iraqi Scud missile.
- The Scud struck an American Army barracks, killing 28 soldiers and injuring around 100 other people.
- The system's internal clock was multiplied by 1/10 to produce the time in seconds.
- The binary expansion of 1/10 is 0.0001100110011001100110011001100.... It is not representable precisely as a binary number.

An Example: Patriot Missile continued...

- The 24 bit register in the Patriot stored instead 0.00011001100110011001100
- An error of 0.0000000000000000000000011001100... binary, or about 0.000000095 decimal.
- The unit had been running for approximately 100 hours so the internal clock was about 0.34 of a second out (a scud covers around 500 metres in that time).
- The disagreement between the Patriot's clock and the radar clock was sufficient that the missile could not be detected.
- The problem was exacerbated by poor maintenance.

An Example: Patriot Missile...conclusions

- Containing coding errors is very hard
  - seemingly insignificant errors result in major changes in behaviour.
- Original fix suggested a change in procedures
  - reboot every 30 hours – impractical in operation.
- Patriot is atypical
  - coding bugs rarely cause accidents alone.
- Maintenance failure
  - failure of coding standards and traceability.

Other Case Studies - Readings

- Ian Sommerville. Software Engineering Case Studies, 2004.
  - The Ariane 5 Launcher Failure
  - The London Ambulance fiasco
  - Airbus Flight Control System

    [web link from the SEOC1 webpage]
- Medical Devices: The Therac-25
  - Nancy Leveson. Safeware: System Safety and Computers. Addison-Wesley, 1995.

    [web link from the SEOC1 webpage]

Software Engineering

- We will study the following areas:
  - Software Requirements: the activities involved in gaining an accurate idea of what the users of the system want it to do.
  - Software Design: the design of a system to meet the requirements.
  - Software Construction: the realisation of the design as a program.
  - Software Testing: the process of checking the code meets the design …
  - Software Configuration, Operation and Maintenance: major cost in the lifetime of systems
- These are the essential activities
- How we deploy effort and arrange these activities is part of Software Engineering Process

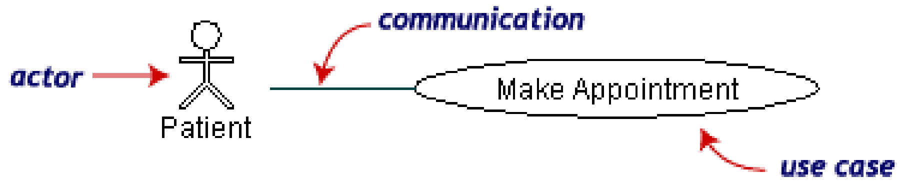
References

- Software Engineering
  - Ian Sommerville. Software Engineering. 7th Edition, Addison-Wesley, 2004.
- Safety-critical Systems
  - Nancy G. Leveson. Safeware: System Safety and Computers. Addison-Wesley, 1995
  - Neil Story. Safety-Critical Computer Systems. Addison-Wesley, 1996.

Models supporting SE

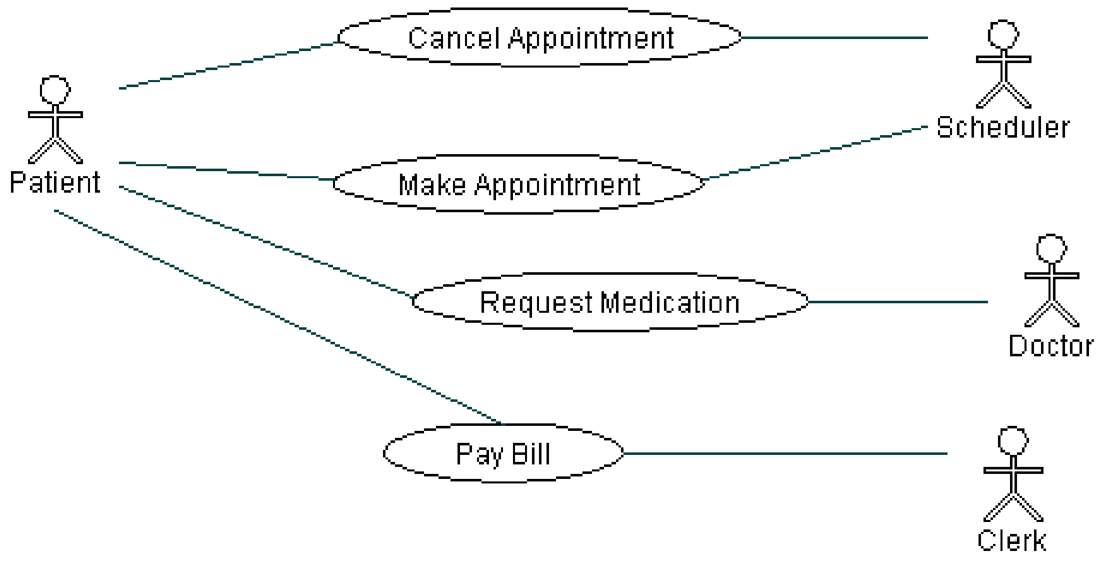
- UML provides a range of graphical notations that capture various aspects of the engineering process.
- Provides a common notation for various different facets of systems.
- Provides the basis for a range of consistency checks, validation and verification procedures.
- Provides a common set of languages and notations that are the basis for creating tools.

UML: Use Case Diagrams



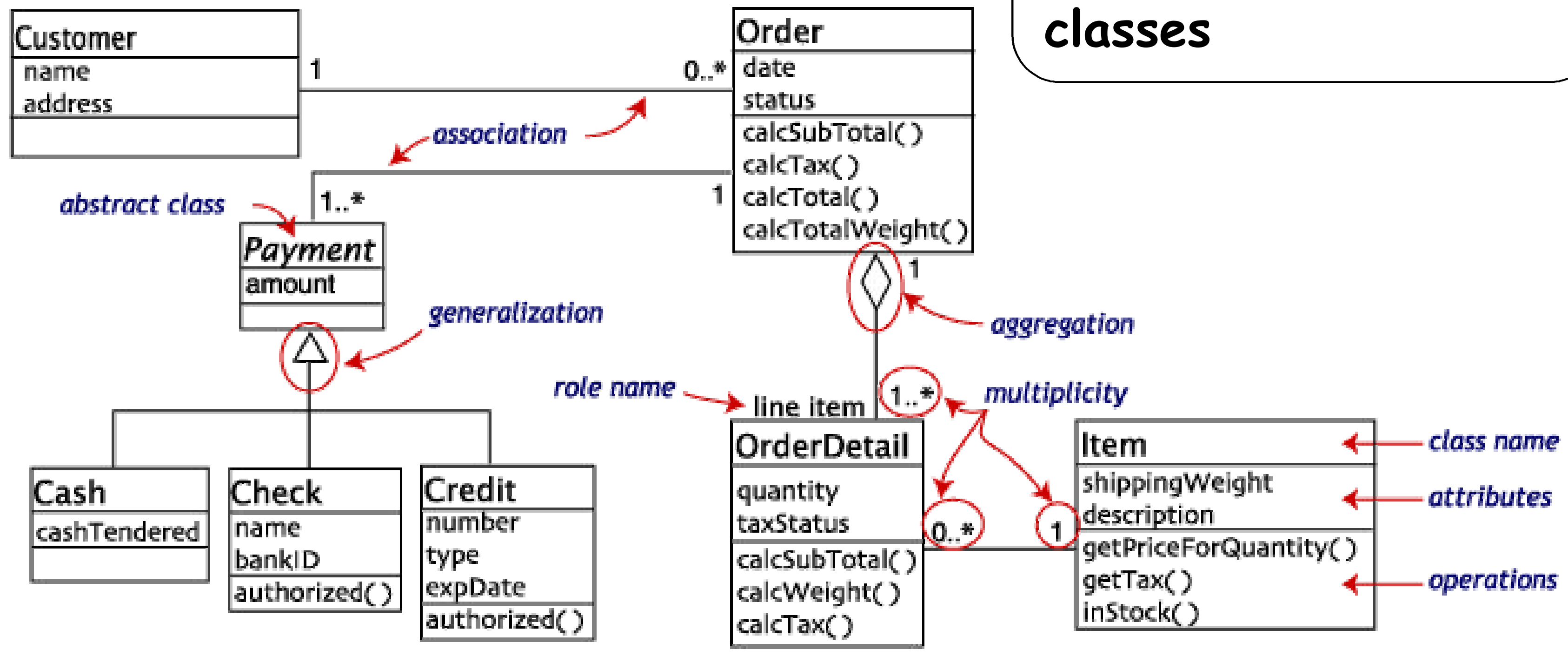
Used to support requirements Capture and analysis

Show the actors' Involvement in System activities



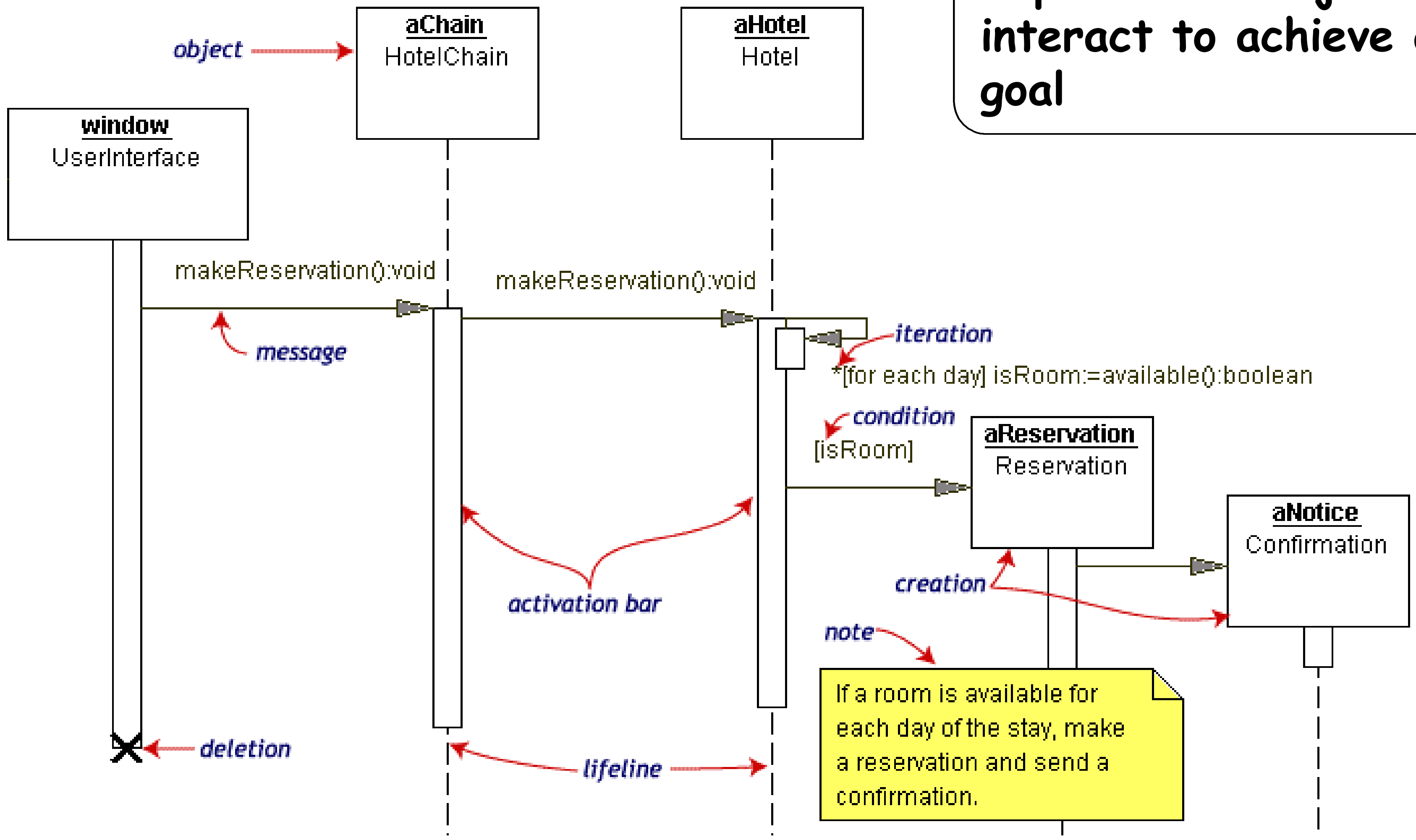
UML: Class Diagrams

Capture the static structure of systems associations between classes

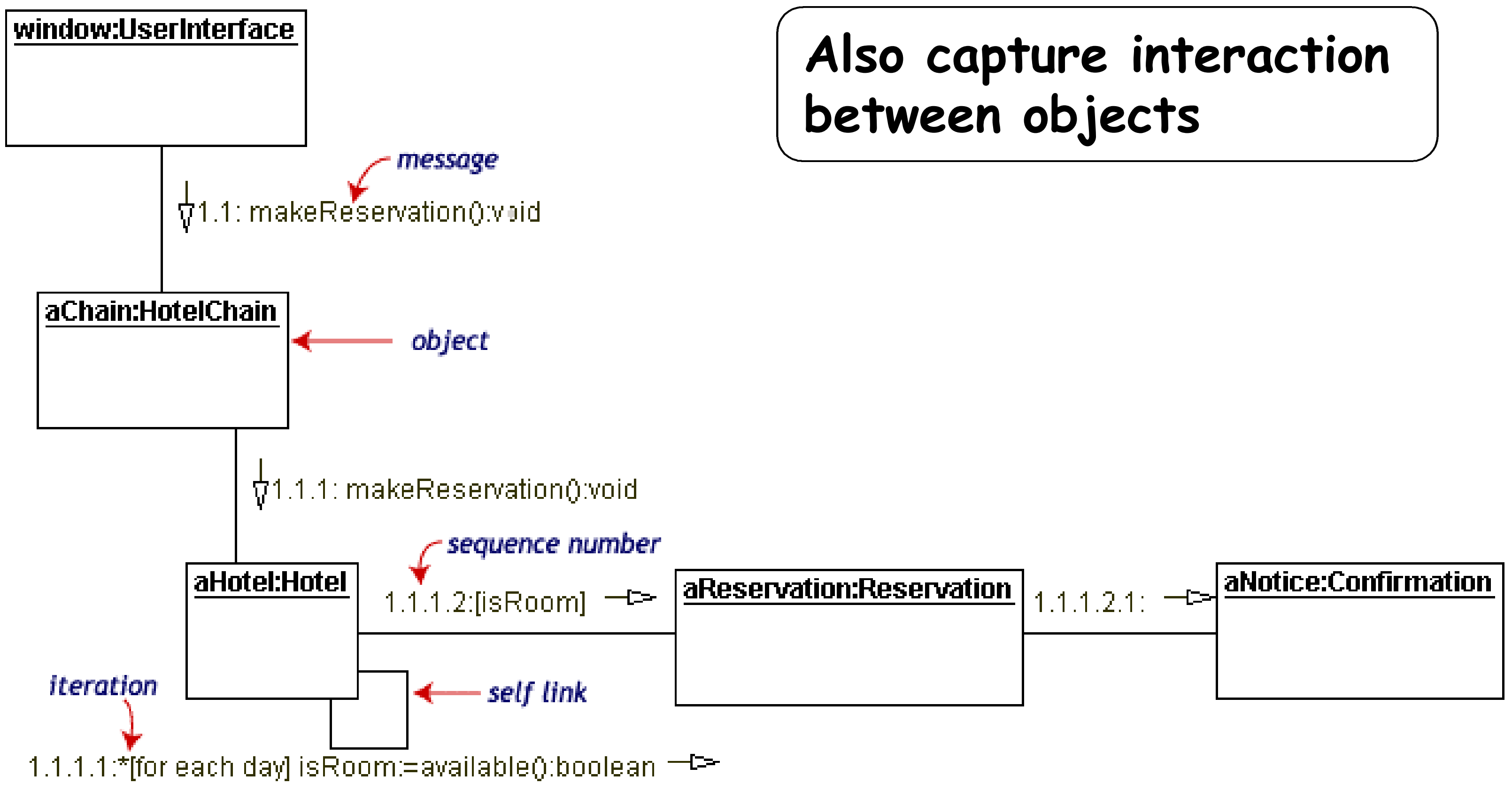


UML: Sequence Diagrams

Capture how objects interact to achieve a goal



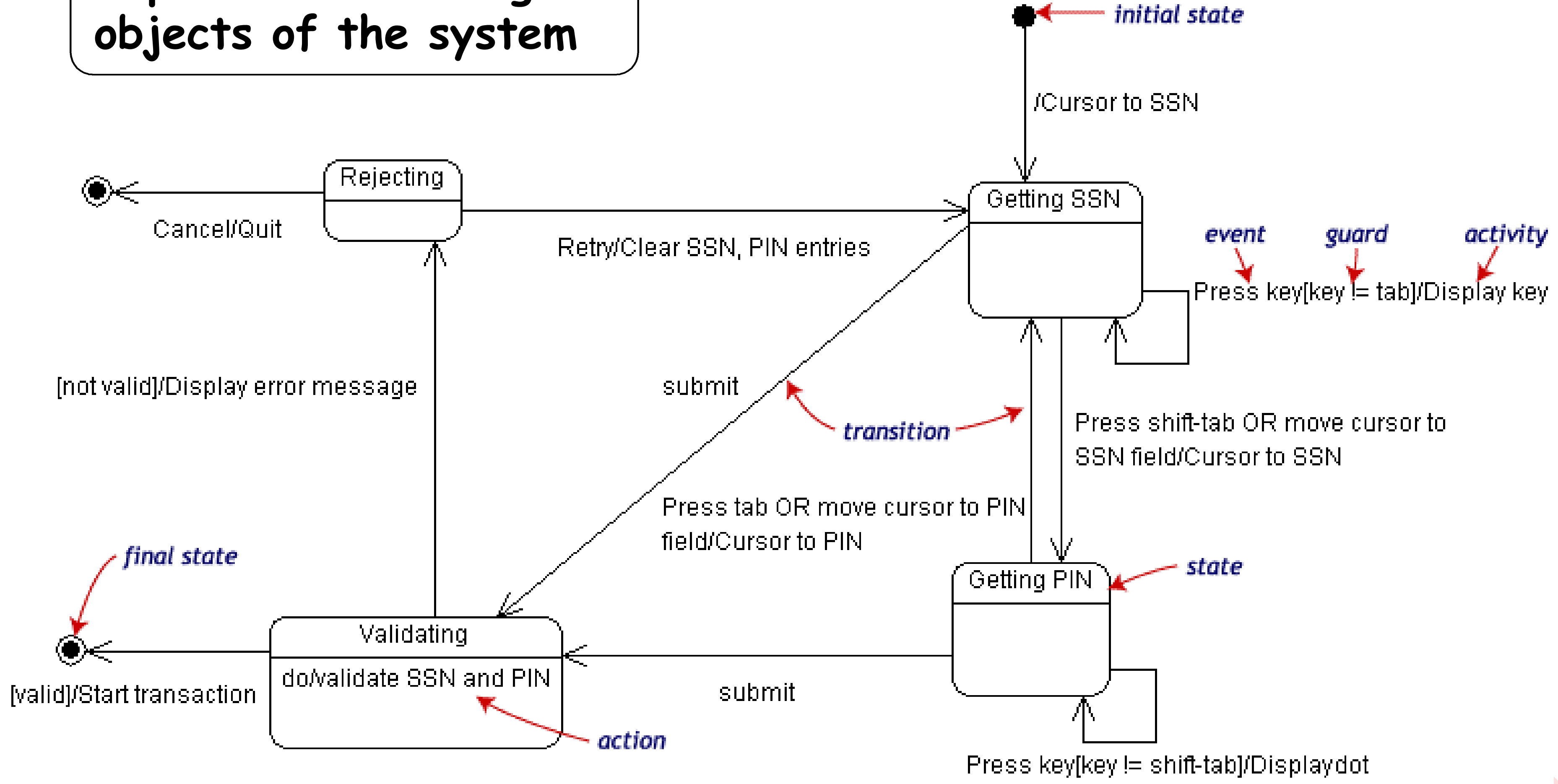
UML: Collaboration Diagrams



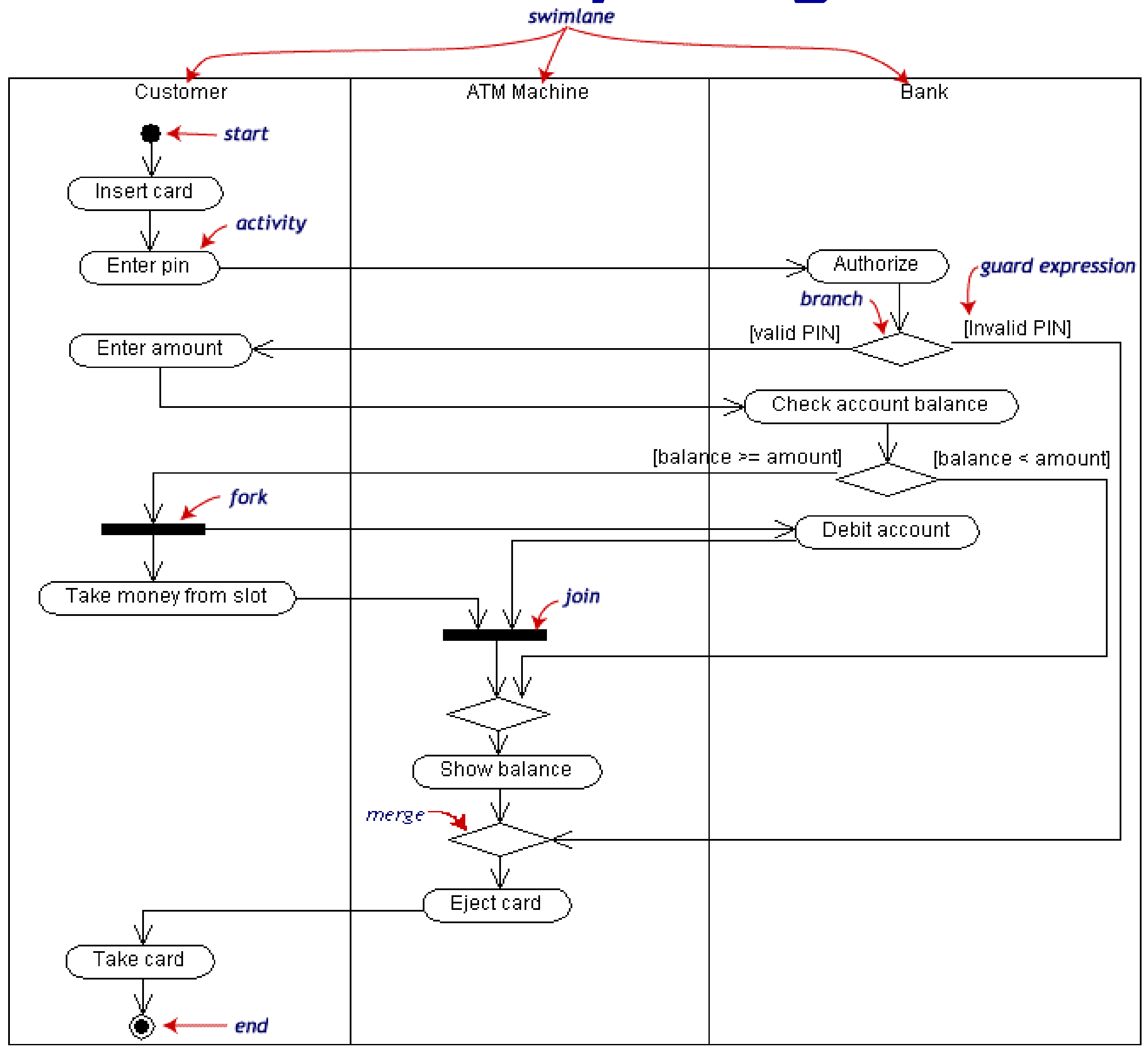
Also capture interaction between objects

UML: Statechart Diagrams

Capture state change in objects of the system

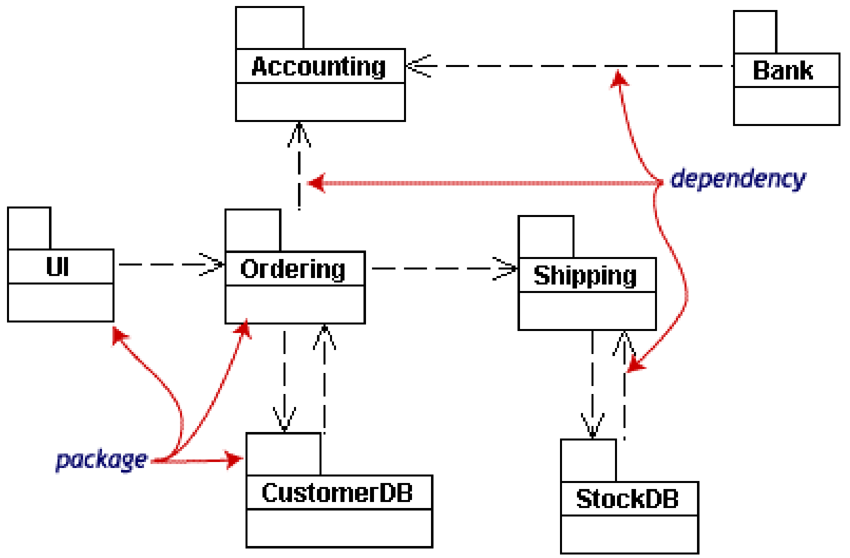

initial state
/Cursor to SSN
Rejecting
Getting SSN
Cancel/Quit
Retry/Clear SSN, PIN entries
event
guard
activity
Press key[key != tab]/Display key
[not valid]/Display error message
submit
transition
Press shift-tab OR move cursor to SSN field/Cursor to SSN
Press tab OR move cursor to PIN field/Cursor to PIN
final state
Getting PIN
state
Validating
do/validate SSN and PIN
[valid]/Start transaction
submit
action
Press key[key != shift-tab]/Displaydot

UML: Activity Diagrams

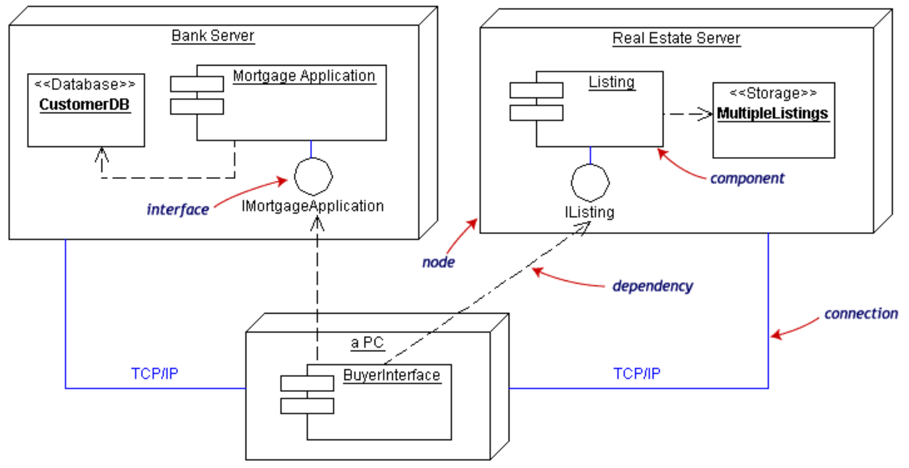


Capture the workflow in a situation

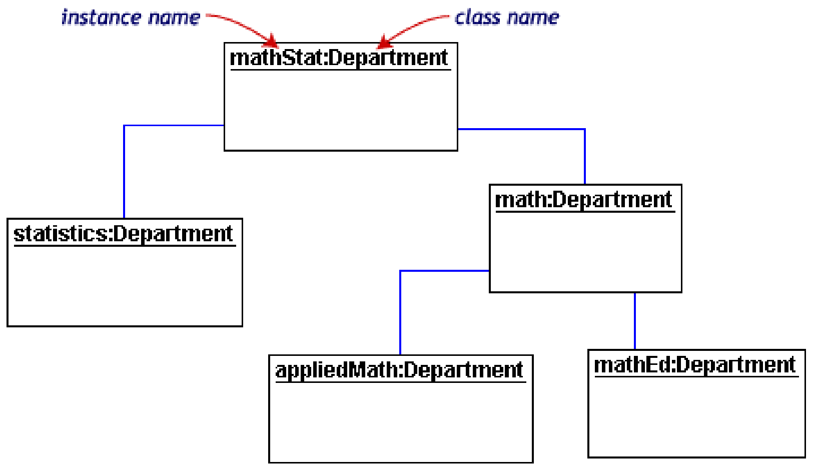
UML: Other Diagrams

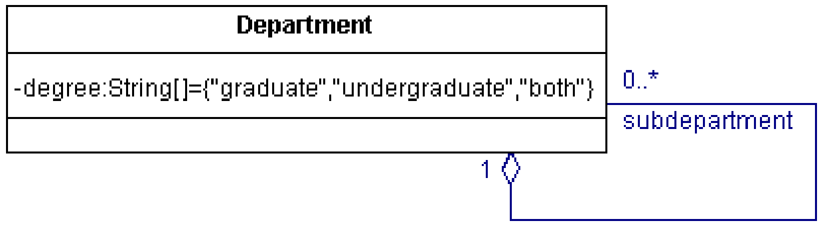


Package



Component and Deployment





Object

Things to Do

- Read the introduction to UML referenced off the course resource page
- Buy the main course book:
  - UML, Schaum's Outline Series, Simon Bennett, John Skelton and Ken Lunn, McGraw-Hill, 2001, ISBN 0-07-709673-8.
- Read chapters 1 and 2 of the UML book
- Read the other software engineering case studies
- Look at the practical page off the course page