College of Information Studies

University of Maryland Hornbake Library Building College Park, MD 20742-4345

Distributed Teams

Week 13

INFM 603

Agenda

- Distributed teams
- Project presentation prep
- Final exam prep

Strategic Choices

- Acquisition
  - Proprietary (“COTS”)
  - Open source
- Implementation
  - Integrate “Best-of-breed” systems
  - “One-off” custom solution

Global Software Development

Barriers

- Geographic distance
- Temporal distance
- Linguistic & cultural distance
- Fear and trust
- Organizational structure
- Process
- Infrastructure
- Project Architecture

Solutions

- Cultural ambassadors
- Configuration management
- Face to face kickoffs
- Modularity
- Well defined interfaces
- Effective handoffs
- Win-win strategies

Extreme Programming

- Planning game
- Customer involvement
- Coding standards
- Simplicity of design
- Pair programming
- Continuous integration
- Refactoring
- Small functional releases
- Collective ownership
- Sustainable pacing
- Metaphor

Open Source “Pros”

- More eyes ⇒ fewer bugs
- Iterative releases ⇒ rapid bug fixes
- Rich community ⇒ more ideas
  - Coders, testers, debuggers, users
- Distributed by developers ⇒ truth in advertising
- Open data formats ⇒ Easier integration
- Standardized licenses

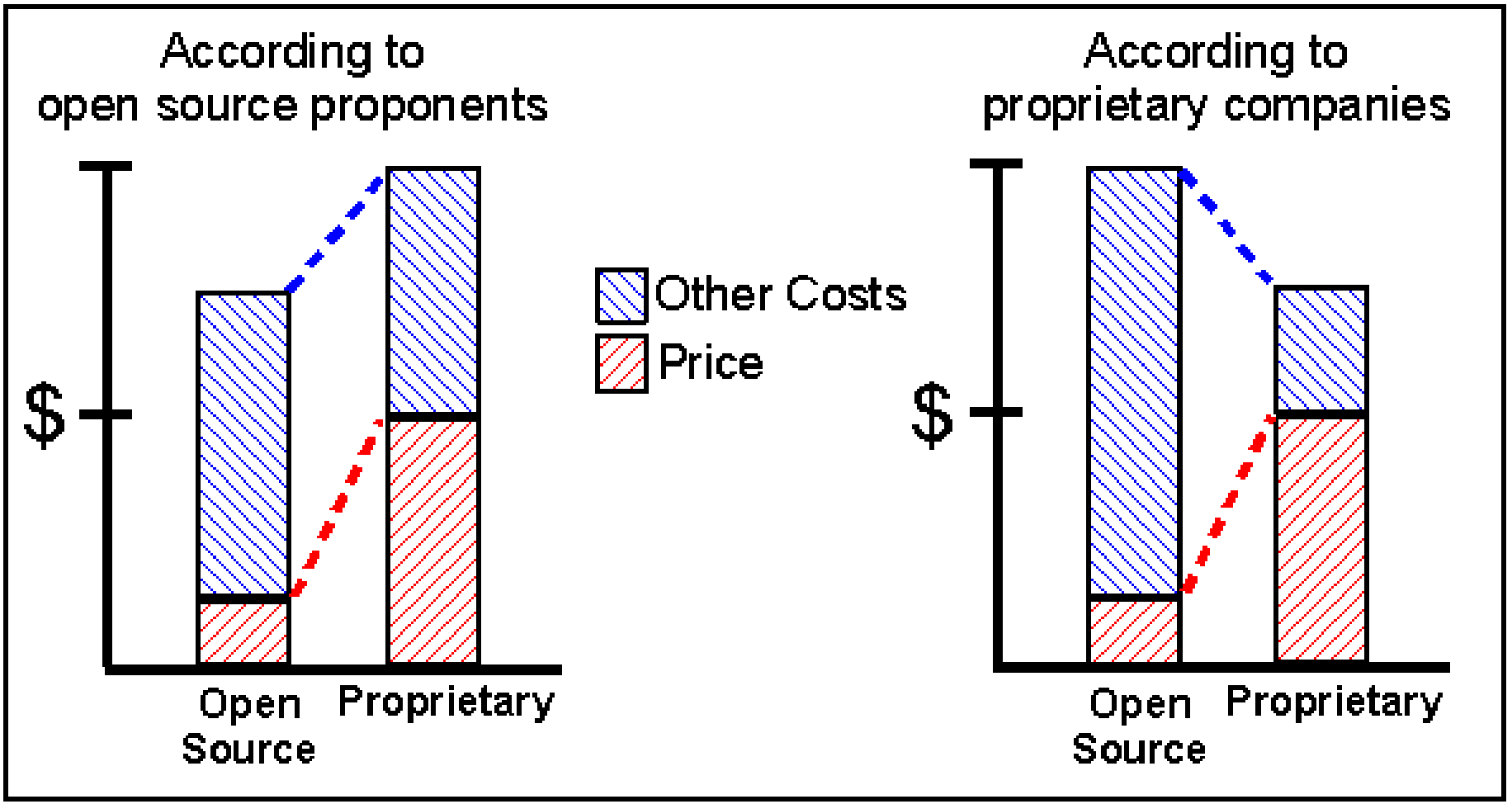
Open Source “Cons”

- Communities require incentives
  - Much open source development is underwritten
- Developers are calling the shots
  - Can result in feature explosion
- Proliferation of “orphans”
- Diffused accountability
  - Who would you sue?
- Fragmentation
  - “Forking” may lead to competing versions
- Little control over schedule

Total Cost of Ownership

- Planning
- Installation
  - Facilities, hardware, software, integration, migration, disruption
- Training
  - System staff, operations staff, end users
- Operations
  - System staff, support contracts, outages, recovery, …

Total Cost of Ownership



Open Source Business Models

- Support Sellers

  Sell distribution, branding, and after-sale services.

- Loss Leader

  Give away the software to make a market for proprietary software.

- Widget Frosting

  If you're in the hardware business, giving away software doesn't hurt.

- Accessorizing

  Sell accessories:
  books, compatible hardware, complete systems with pre-installed software

Project Presentations

- Maximum of 25 minutes
- Goals (from the user's perspective)
- Demo
- Task division between partners
- Most interesting implementation details
- Complete list of limitations
- Lessons learned
- Project process improvement (optional)

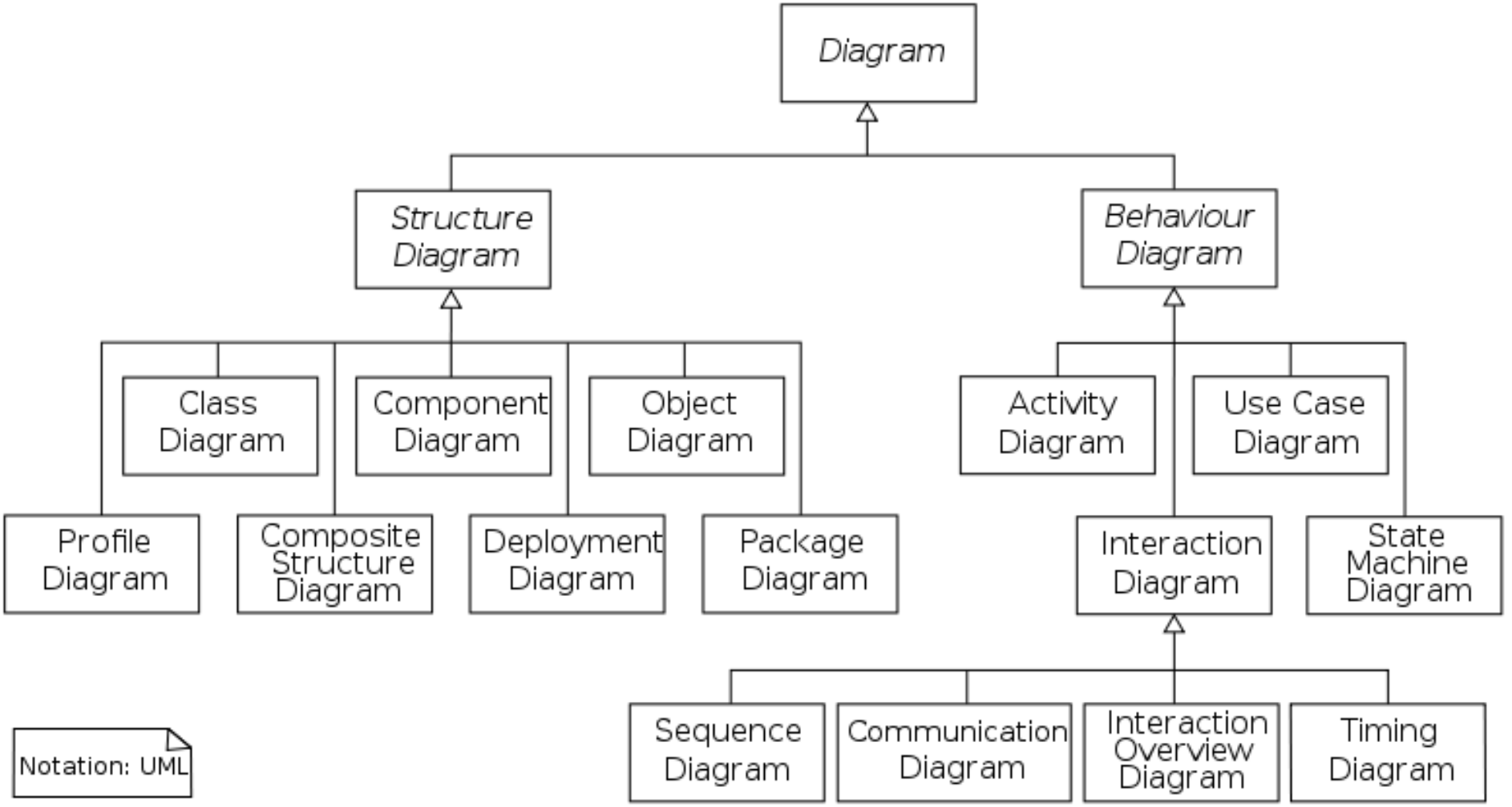
Final Exam

- 2 hours
  - Starts at 6:00 sharp (be early)
  - Ends at 8:00 sharp
- Take it anywhere
  - Classroom will be available
  - Ask me questions by email or phone
- Open everything
  - But **no** communication with **any** other person
- Available from the Web and by email
- Submitted to me by email

Unified Modeling Language

- Real systems are more complex than anyone can comprehend
- Key idea: Progressive refinement
  - Carve the problem into pieces
  - Carve each piece into smaller pieces
  - When the pieces are small enough, code them
- UML provides a formalism for doing this
  - But it does not provide the process

Unified Modeling Language



Specifying Structure

- Capturing the big picture
  - Use case diagram (interactions with the world)
  - Narrative
  - Scenarios (examples to provoke thinking)
- Designing the object structure
  - Class diagram (“entity-relationship” diagram)
  - Object diagram (used to show examples)

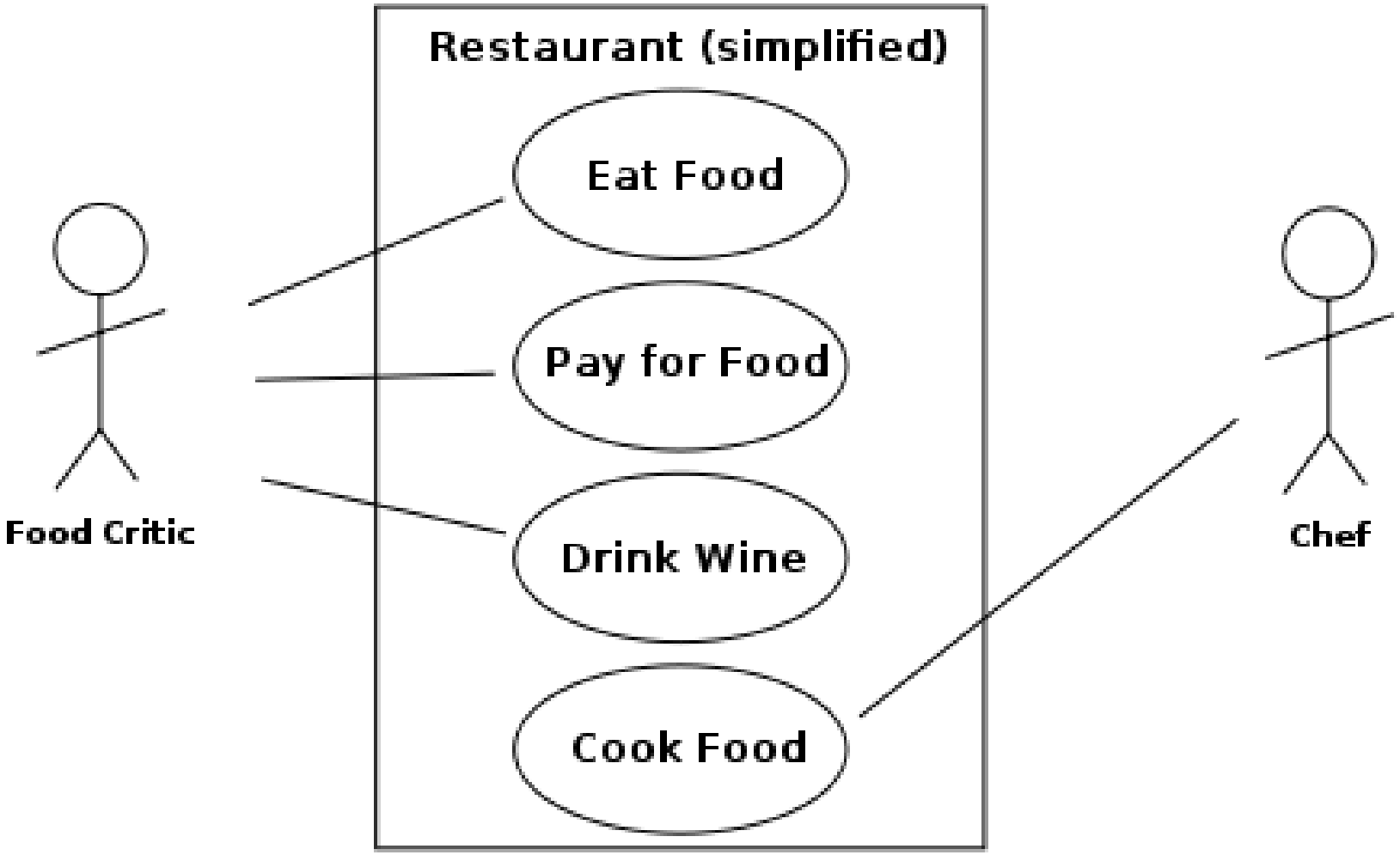
Specifying Behavior

- Represent a candidate workflow
  - Activity diagram (a “flowchart”)
- Represent object interactions for a scenario
  - Collaboration diagram (object-based depiction)
  - Sequence diagram (time-based depiction)
- Represent event-object interactions
  - Statechart diagram (a “finite state machine”)

Use Case Design

- Use Case Diagram
  - Input-output behavior
- Use Case Narrative
  - Explains each use case
- Use Case Scenario
  - Activity diagram shows how the use cases are used together

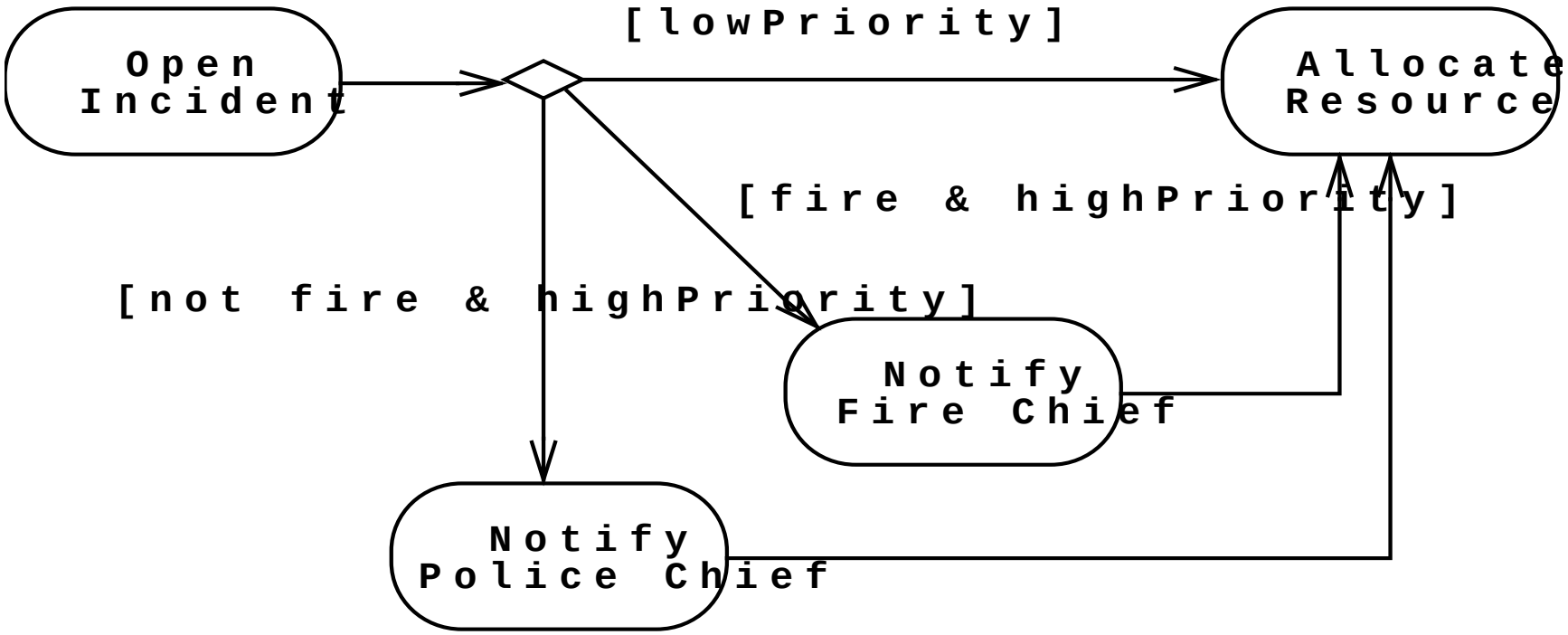
Use Case Diagram



Use Case Diagram

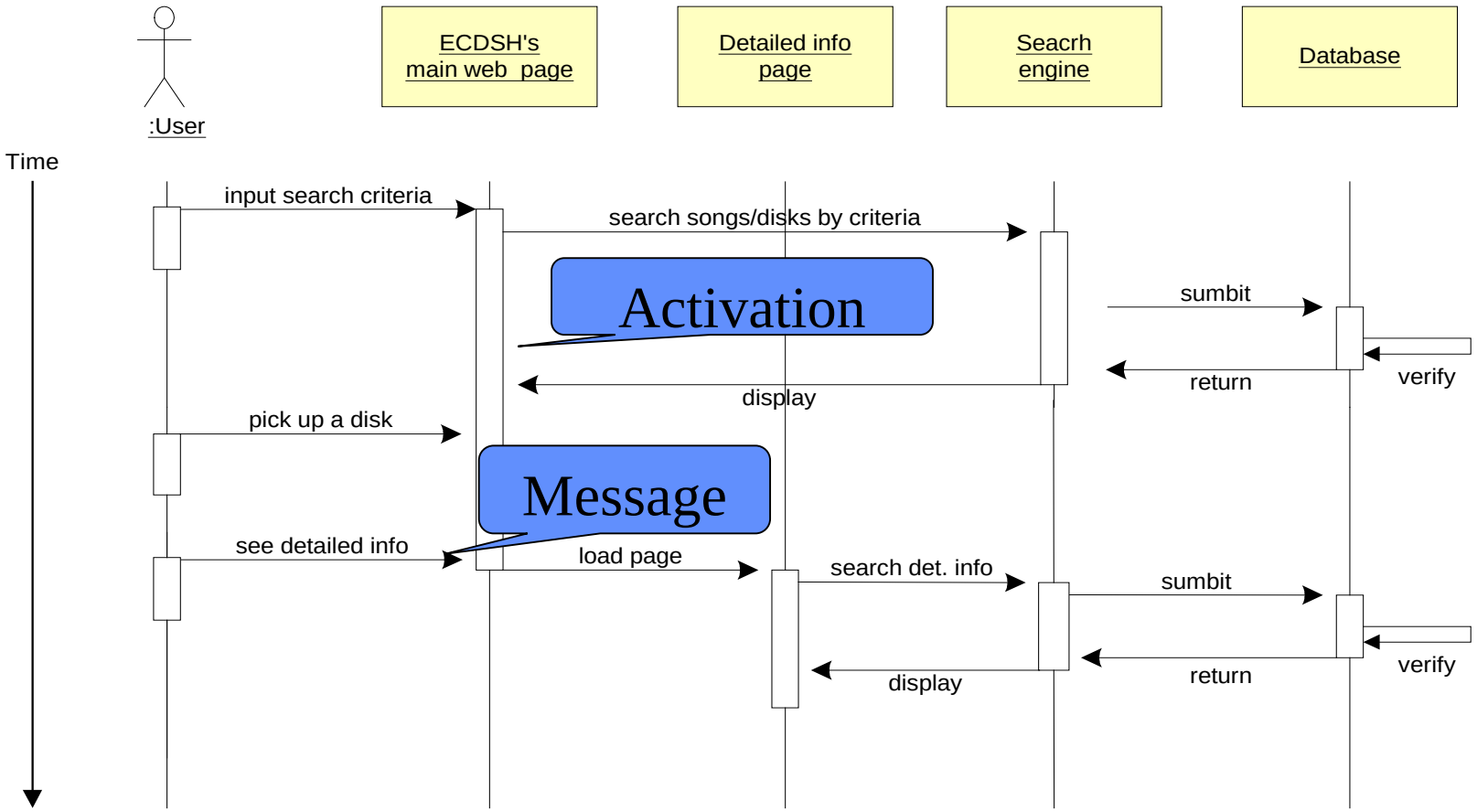
- External “actors”
  - Roles of people
  - Types of systems
- Use cases
  - Top-level functions (solid arrows to/from actors)
- Relationships among use cases
  - Always-depends-on (dashed <<include>>)
  - Sometimes-is-depended-on (dashed <<extend>>)
  - Inherits-from (solid triangle-arrow)

Activity Diagram: Modeling Decisions



Thanks to Satish Mishra

Sequence Diagram



Thanks to Satish Mishra

Good Uses for UML

- Focusing your attention
  - Design from the outside in
- Representing partial understanding
  - Says what you know, silent otherwise
- Validate that understanding
  - Structuring communication with stakeholders

Avoiding UML Pitfalls

- Don’t sweat the notation too much
  - The key is to be clear about what you mean!
- Don’t try to make massive conceptual leaps
  - Leverage encapsulation to support abstraction
- Don’t get to attached to your first design
  - Goal is to find weaknesses in your understanding