Inception, Requirements and Activity Diagrams

Software Design and Analysis

CSCI 2040, Jarek Szlichta

Objectives

- Inception
- Understanding requirements
- Making UML activity diagrams
  - Activity diagram notation
  - Activity diagram execution model
- To know when and how to use activity diagrams

Inception

Inception

- Inception is the initial short step to establish a common vision and basic scope for the project
  - **Inception is Not the Requirements Phase**
- It includes analysis
  - of perhaps 10% of the use cases
  - analysis of the critical non-functional requirements creation of a business case
  - preparation of the development environment so that programming can start in the following elaboration phase

What is Inception?

- Most projects require a short initial step in which the following kinds of questions are explored:
  - What is the vision and business case for this project?
  - Feasible?
  - Buy and/or build?
  - *Rough* unreliable range of cost: Is it $10K, 100K or in the millions?
  - Should we proceed or stop?
- The inception phase should be relatively short for most projects, such as one or a few weeks long.

Does this Analogy Help?

- In the oil business, when a new field is being considered, some of the steps include:
  1. Decide if there is enough evidence or a business case to even justify exploratory drilling.
  2. If so, do measurements and exploratory drilling.
  3. Provide scope and estimate information.
  4. Further steps…

Does this Analogy Help?

- In the oil business, when a new field is being considered, some of the steps include:
  1. Decide if there is enough evidence or a business case to even justify exploratory drilling.
  2. If so, do measurements and exploratory drilling.
  3. Provide scope and estimate information.
  4. Further steps…

Does this Analogy Help?

- In the oil business, when a new field is being considered, some of the steps include:
  1. Decide if there is enough evidence or a business case to even justify exploratory drilling.
  2. If so, do measurements and exploratory drilling.
  3. Provide scope and estimate information.
  4. Further steps…
- The inception phase is like step one in this analogy.

Sample Inception Artifacts

Artifact	Comment
Vision and Business Case	Describes the high-level goals and constraints, the business case, and provides an executive summary.
Use-Case Model	Describes the functional requirements. During inception, the names of most use cases will be identified, and perhaps 10% of the use cases will be analyzed in detail.
Supplementary Specification	Describes other requirements, mostly non-functional. During inception, it is useful to have some idea of the key non-functional requirements that have will have a major impact on the architecture.
Glossary	Key domain terminology, and data dictionary.
Risk List & Risk Management Plan	Describes the risks (business, technical, resource, schedule) and ideas for their mitigation or response.
Prototypes and proof-of-concepts	To clarify the vision, and validate technical ideas.
Iteration Plan	Describes what to do in the first elaboration iteration.
Phase Plan & Software Development Plan	Low-precision guess for elaboration phase duration and effort. Tools, people, education, and other resources.
Development Case	A description of the customized UP steps and artifacts for this project. In the UP, one always customizes it for the project.

Isn't That a Lot of Documentation?

- Artifacts should be considered optional. Choose to create only those that really add value for the project.
- This is evolutionary development,
  - the point is not to create complete specifications during this phase, but initial, rough documents, that are refined during the elaboration iterations,
    - in response to the invaluable feedback from early programming and testing.
- Artifacts from previous projects can be partially reused on later ones.
  - It is common for there to be many similarities in risk, project management, testing, and environment artifacts across projects.

You know you Didn't Understand the Inception When..?

- It is more than "a few" weeks long for most projects.
- There is an attempt to define most of the requirements.
- Estimates or plans are expected to be reliable.
- You define the architecture (this should be done iteratively in elaboration).
- There is no Business Case or Vision artifact.
- All the use cases were written in detail.
- None of the use cases were written in detail;
  - rather, 10, 20% should be written in detail to obtain some realistic insight into the scope of the problem.

Understanding Requirements

Definition: Requirements

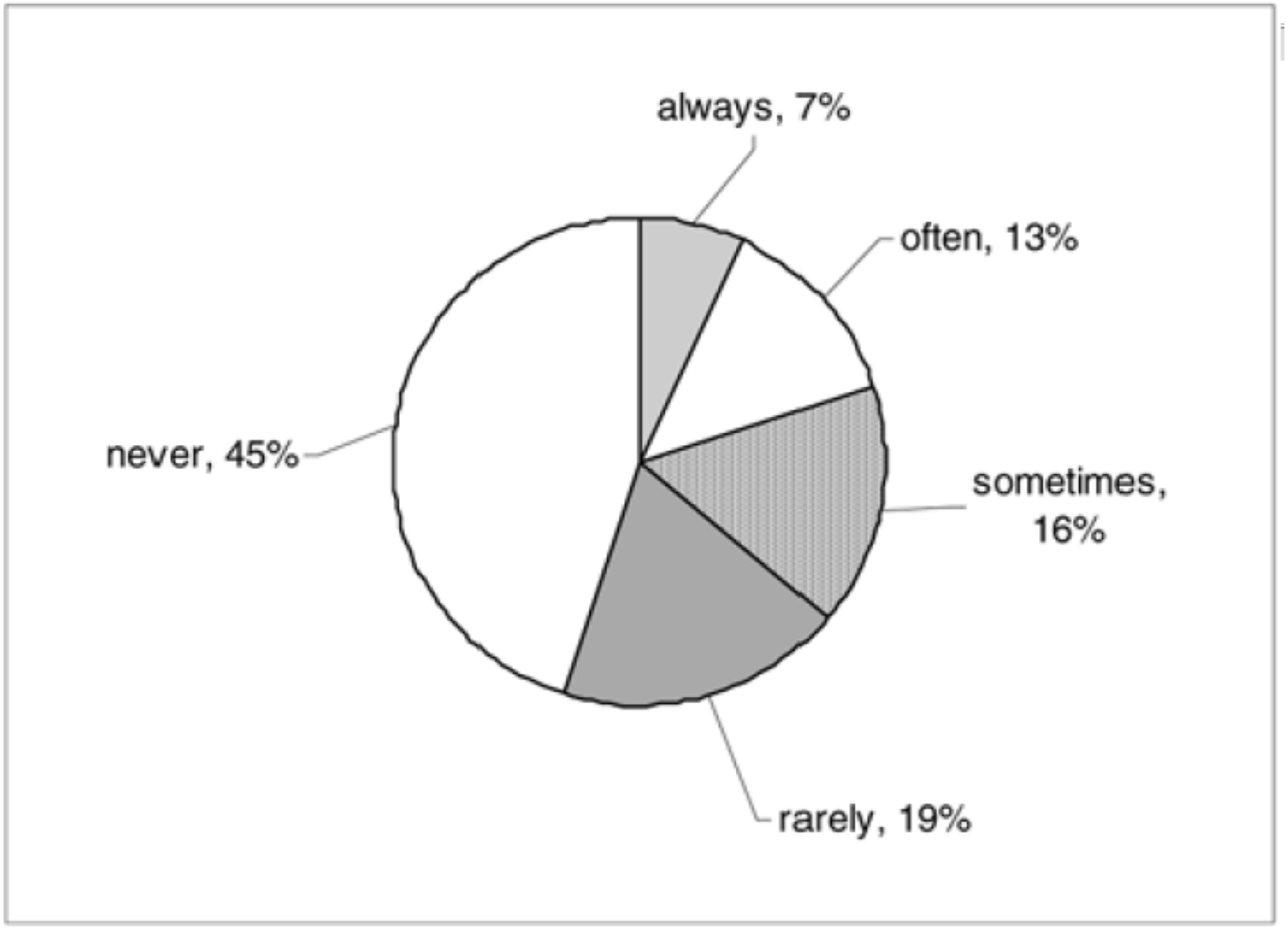
- **Requirements** are capabilities and conditions to which the system must conform.
- A prime challenge of requirements analysis is to
  - find,
  - communicate,
  - and remember (that usually means write down)

  what is really needed, in a form that clearly speaks to the client and development team members.

Evolutionary vs. Waterfall Requirements

- There is still a speculative belief in the efficacy of full, early requirements analysis for software projects (i.e., the waterfall).
  - But on average, 25% of the requirements change on software projects.
- In the **iterative methods** we start production-quality programming and testing long before most of the requirements have been analyzed or specified
  - perhaps when only 10% or 20% of the most architecturally significant, risky, and high business-value requirements have been specified.

Actual Use of Waterfall-specified Features



Iterative and Evolutionary Requirements

- These results don't imply that the solution is to start pounding away at the code near Day One of the project,
  - and forget about requirements analysis.
- There is a middle way: iterative and evolutionary requirements analysis
  - combined with early time-boxed iterative development and frequent stakeholder participation, evaluation, and feedback on partial results.

What are the Types of Requirements?

- **Functional** features, capabilities, security.
- **Usability** help, documentation.
- **Reliability** frequency of failure and recoverability
  Performance response times, availability, resource usage.
- **Supportability** maintainability, internationalization, configurability.

What are the Categories of Requirements?

- In common usage, requirements are categorized as
  - **functional** (behavioral), e.g., display of the online inventory in the online store, choosing shipping methods
  - **non-functional** (everything else), e.g., documentation, price etc;

How are Requirements Organized in UP Artifacts?

- **Use-Case Model** A set of typical scenarios of using a system. They are primarily for functional (behavioral) requirements.
- **Glossary** In its simplest form, the Glossary defines noteworthy terms.
- **Vision** Summarizes the business case for the project.
- **Business Rules -** many applications may need to conform to domain or business rules**.**
  - An excellent example is government tax laws.

UML Activity Diagrams

Processes and Their Description

A **process** is a collection of related tasks

We will introduce a UML process notation called an Activity Diagram

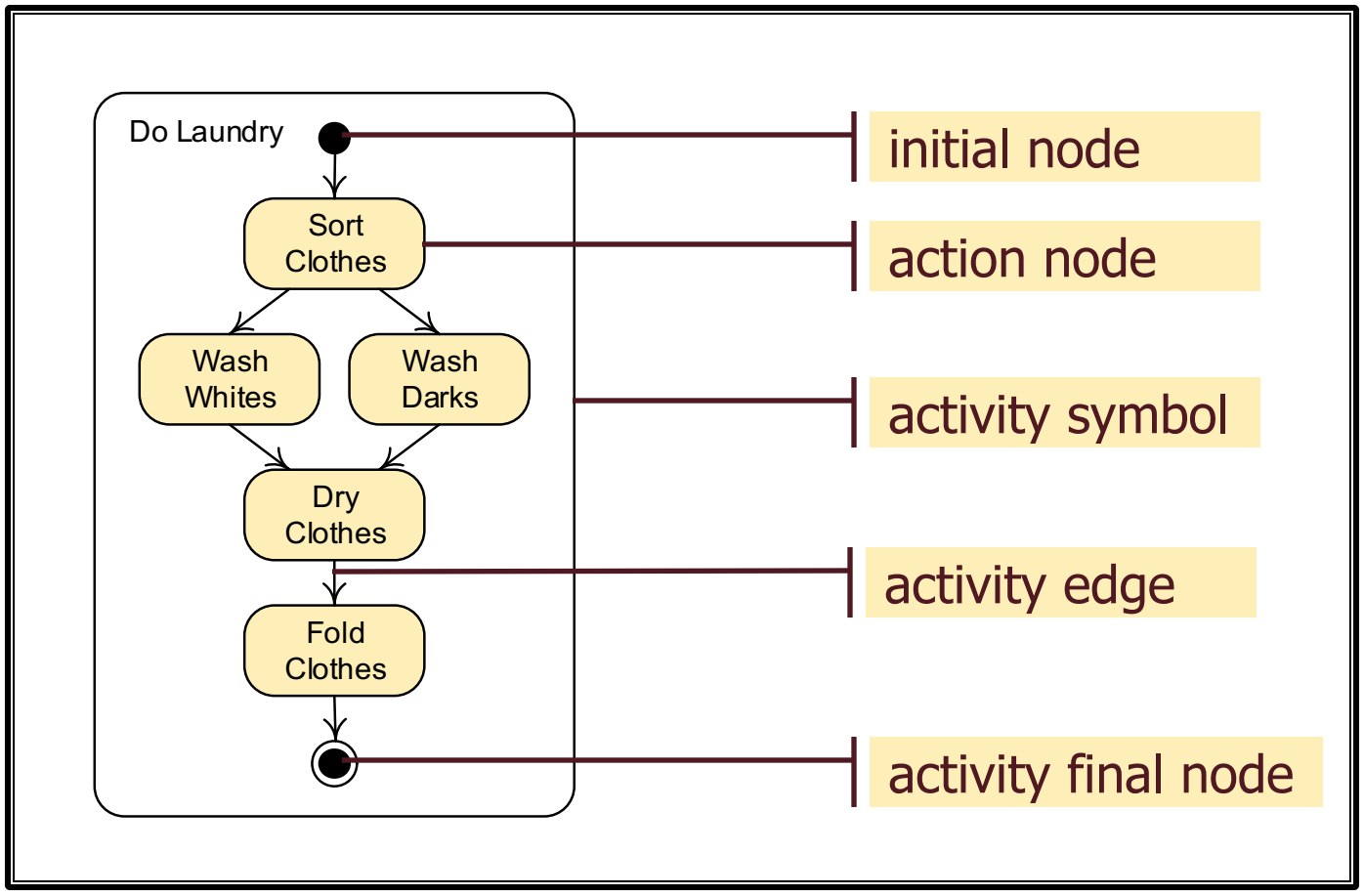
An **activity diagram** shows actions and the flow of control and data between them

Activities and Actions

An **activity** is a non-atomic task decomposable into actions.

An **action** is a task that cannot be broken into parts.

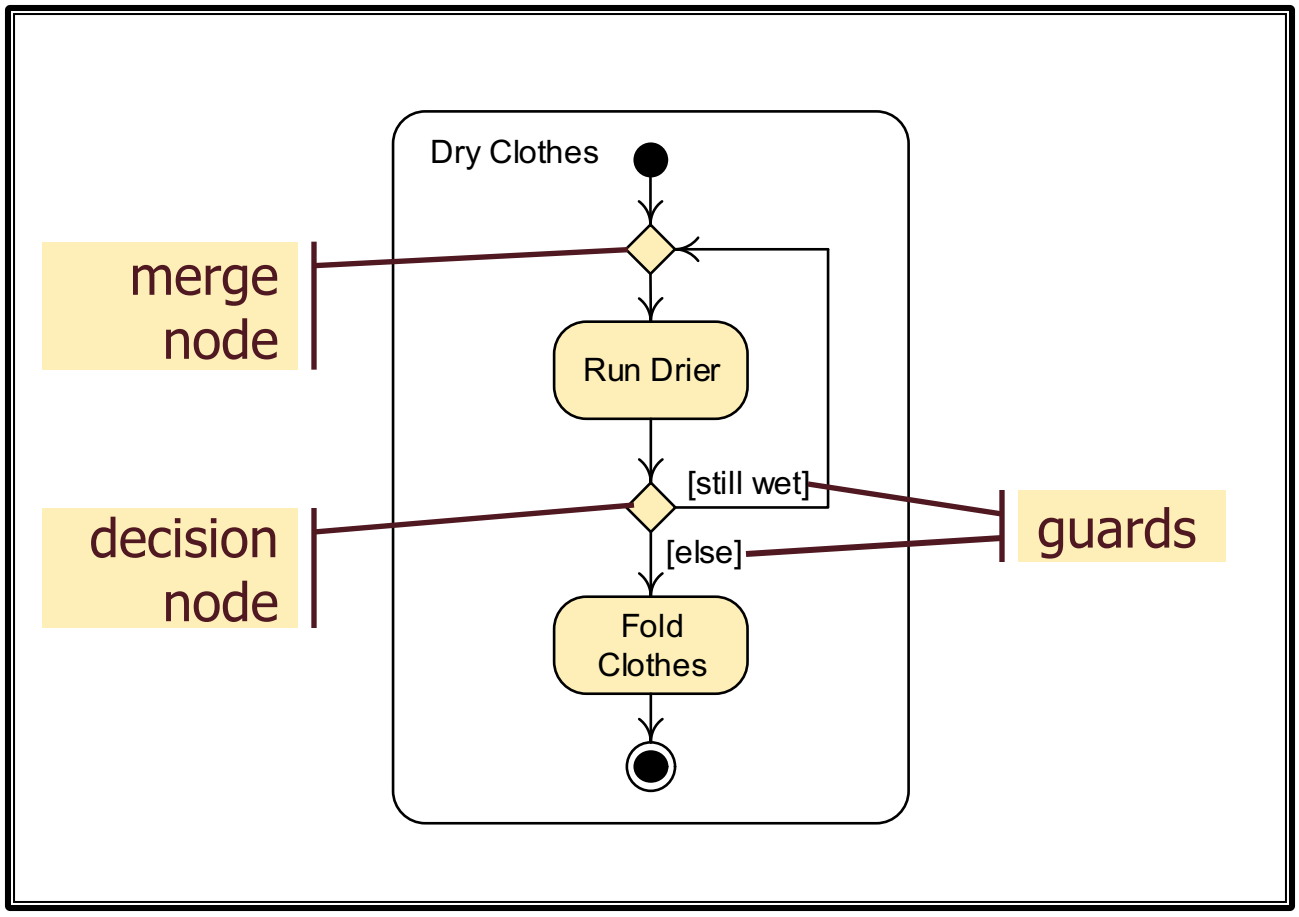
Activity Graph Elements



Execution Model

- Execution is modeled by **tokens**
  - that are produced by action nodes, travel over action edges, and are consumed by action nodes
- When an action node completes execution, it produces tokens on each of its outgoing edges.

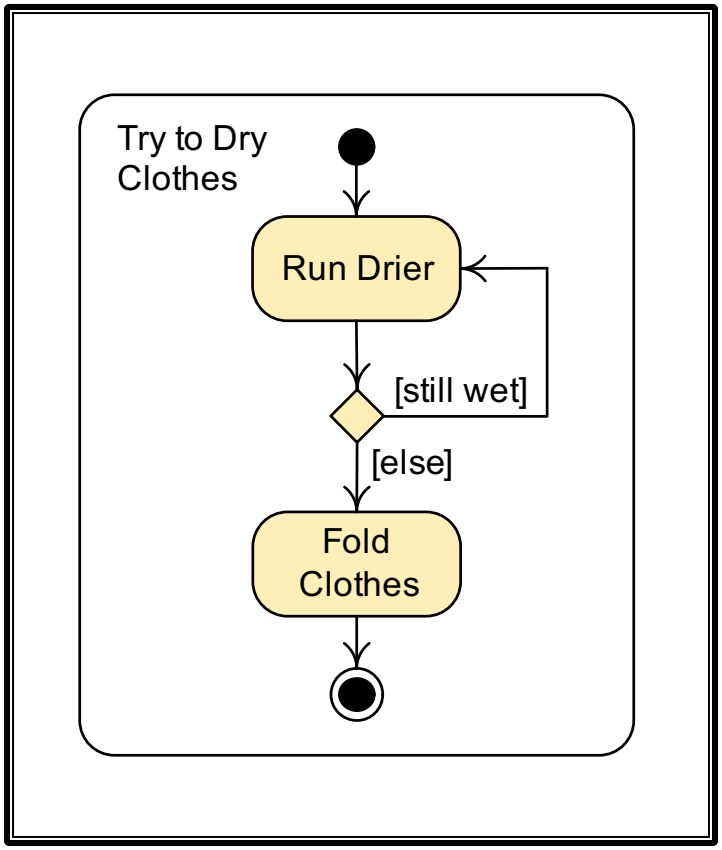
Branching Nodes



Branching Execution

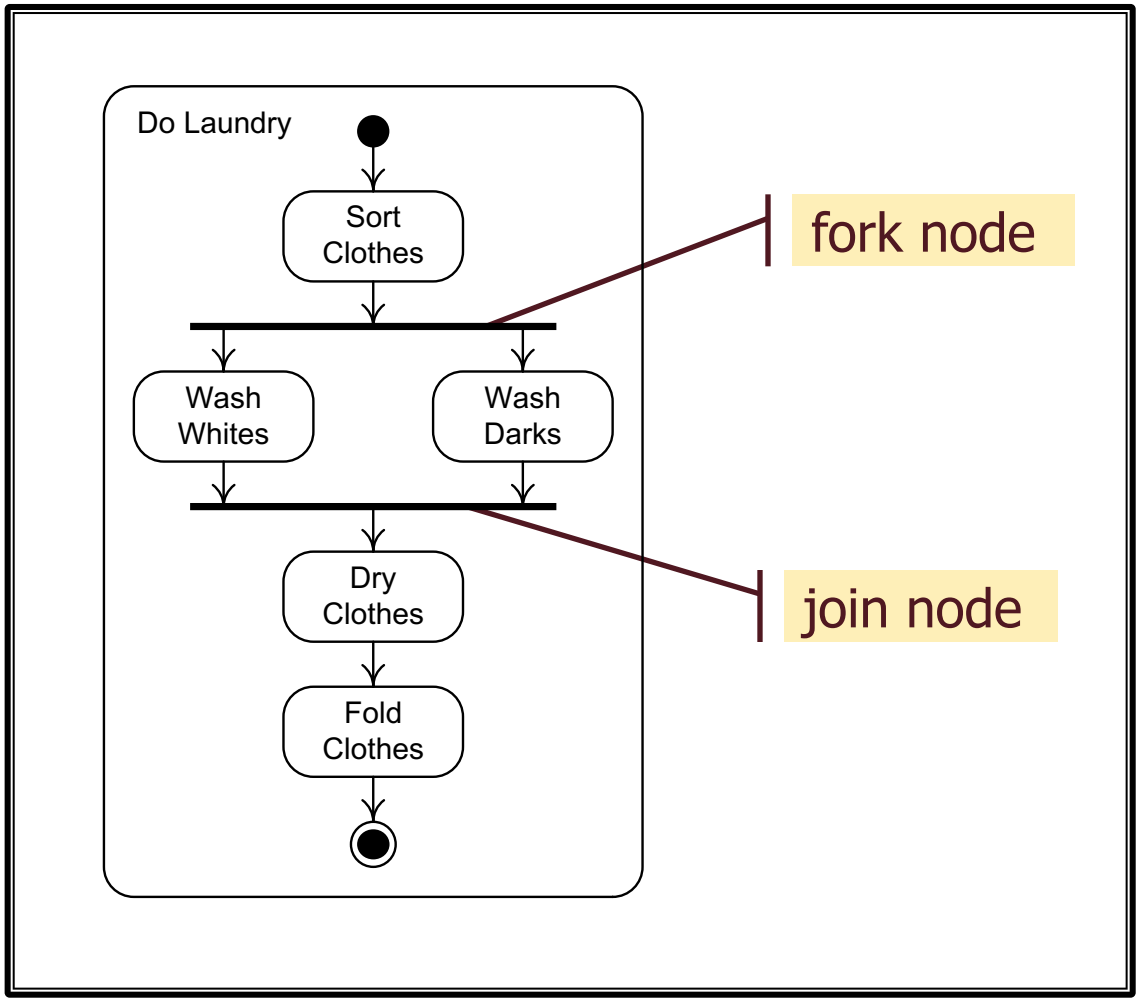
- Token is made available on the outgoing edge whose guard is true.
- Guards must be mutually exclusive!

Incorrect



Merge node has to be used!

Forking and Joining Nodes

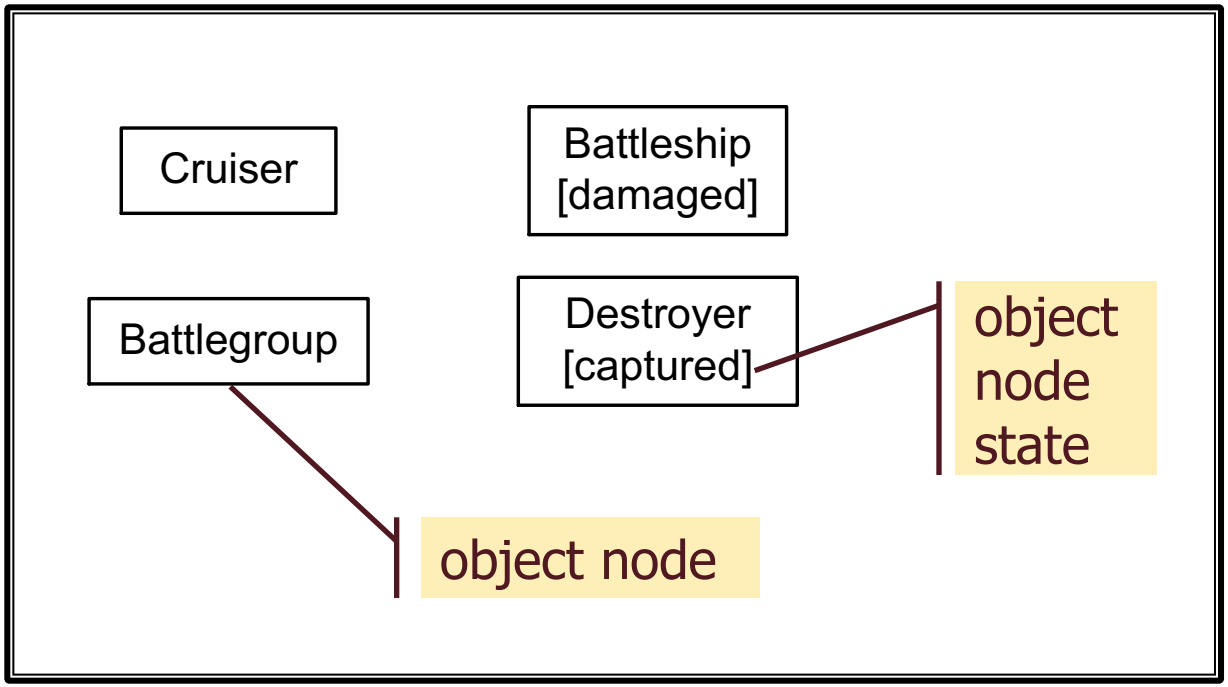


Forking and Joining Execution

- This models concurrency and synchronization.
- To support parallelism
  - Join node waits for both of them to continue

Object Nodes

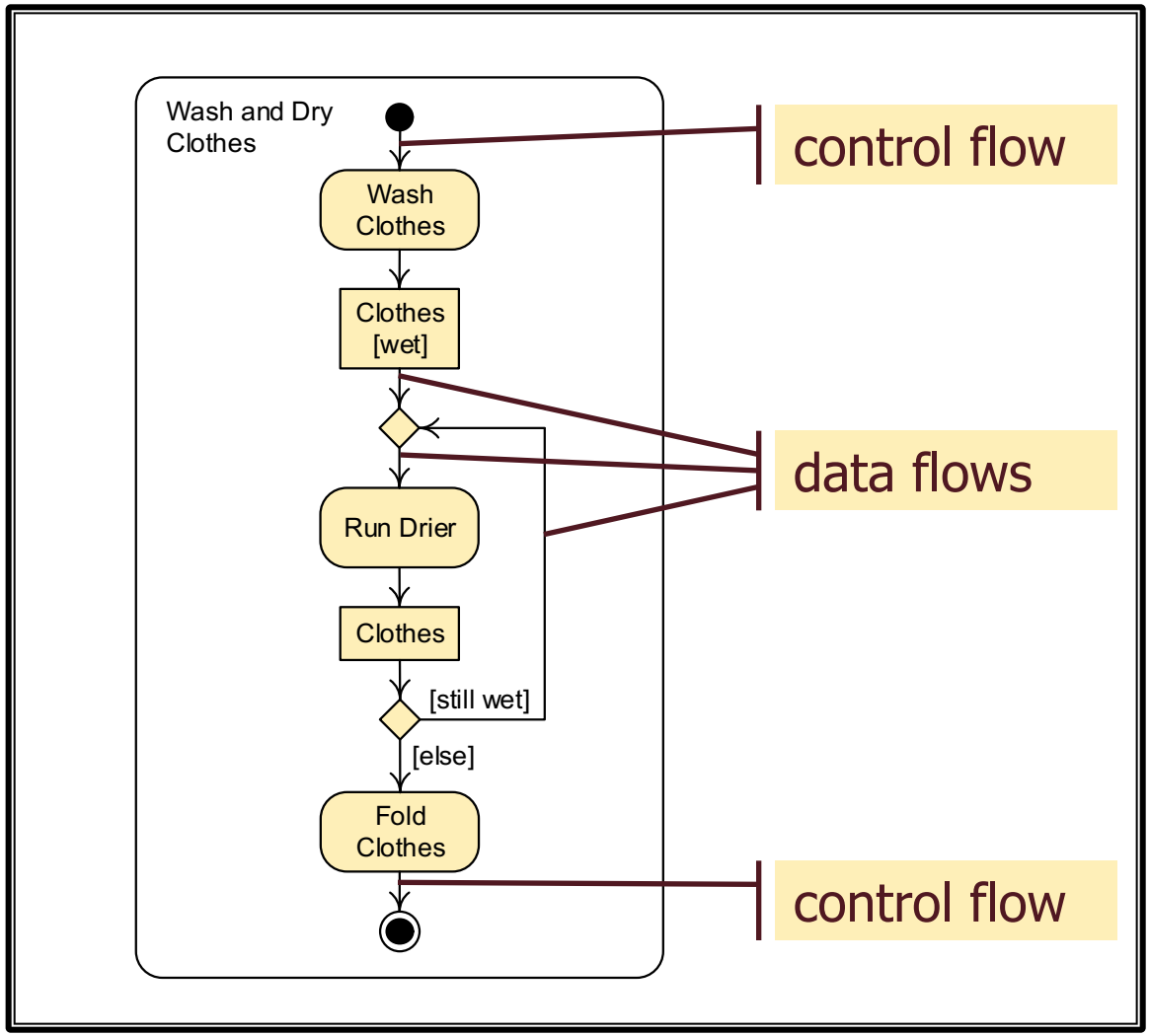
Data are shown as object nodes.



Control and Data Flows

- **Control tokens** do not contain data, **data tokens** do.
- Rules for token flow through nodes apply to both control and data tokens

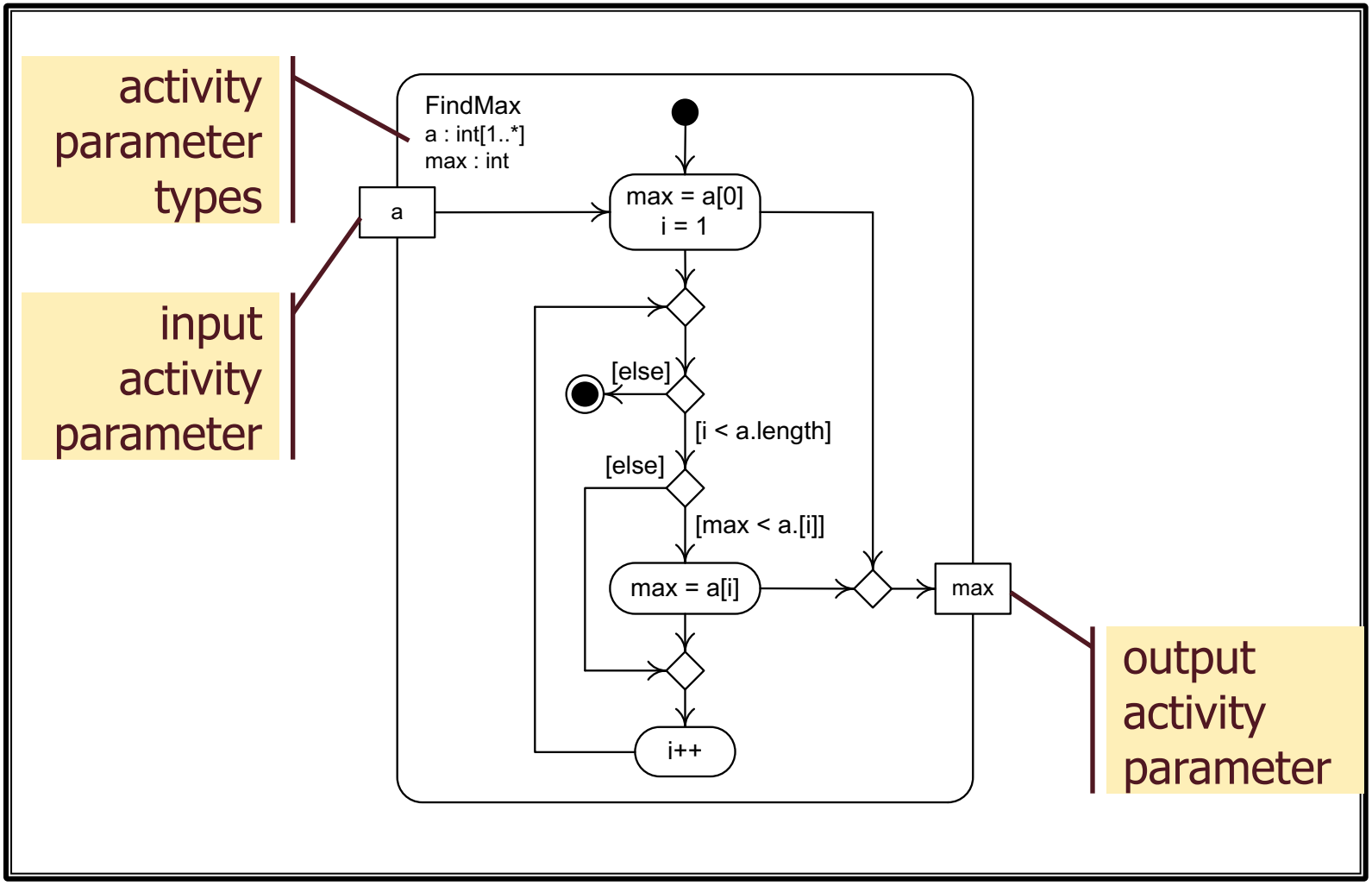
Control and Data Flow Example



Activity Parameters

- Activity parameters are object nodes placed on activity symbol boundaries to indicate inputs or outputs.
- Activity parameter types are specified in the activity symbol

Activity Parameter Example



Activity Diagram Heuristics

- Flow control and objects down the page and left to right.
- Name activities and actions with verb phrases.
- Name object nodes with noun phrases
- Use concurrency when needed
- Use the [else] guard at every branch

When to Use Activity Diagrams

When making a dynamic model of any process.

- Design processes
  - During analysis
  - During design

Summary

- UML activity diagrams model processes by representing actions and the flow of control and data between them.
- Activity symbols contain activity graphs consisting of
  - action nodes
  - action edges
  - data nodes
  - special nodes for starting and stopping activities, branching, forking, and joining
- Activity diagrams can represent any process and are useful throughout software design!

Your Action Items

- Install Microsoft Visio if you have not done it already
  - MSDN DreamSpark Premium
  - https://uoitsci.onthehub.com/WebStore/Security/SignIn.aspx
  - Workshop?
- Review Slides!
- Read chapter 4 and 5 from the textbook