

System Analysis and Design

Modeling

Lecture 09

Mustafa Hasan

Head of CSE



Model

A partial abstract representation of a real-world system

An inexpensive way to analyze, communicate, test, and document our understanding of the system

Type of Models

Computational

- Computer simulations representing time-varying behavior of a system

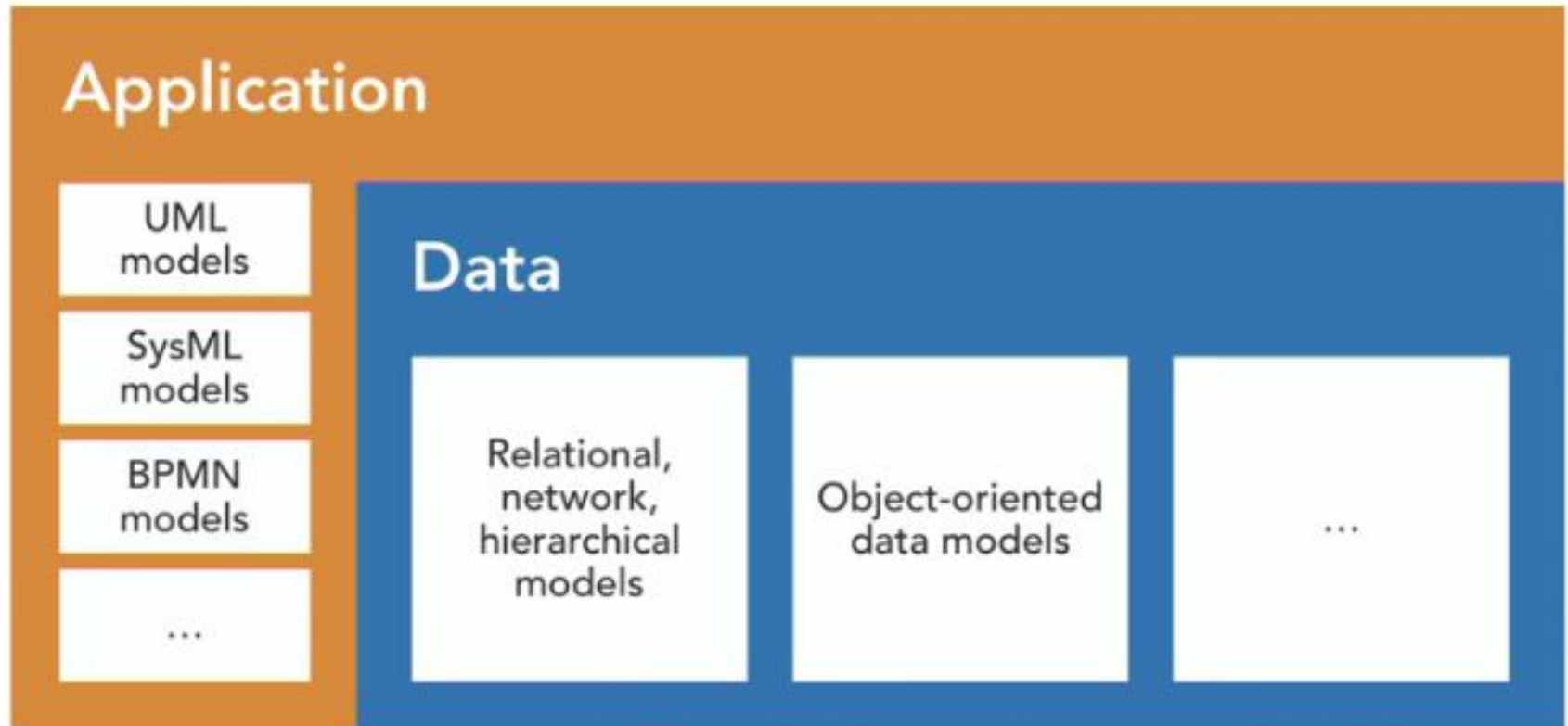
Analytical

- Mathematical models of relationships among variables in a system

Nonanalytical/descriptive

- Describe components and their relationships in a system

Models of Software





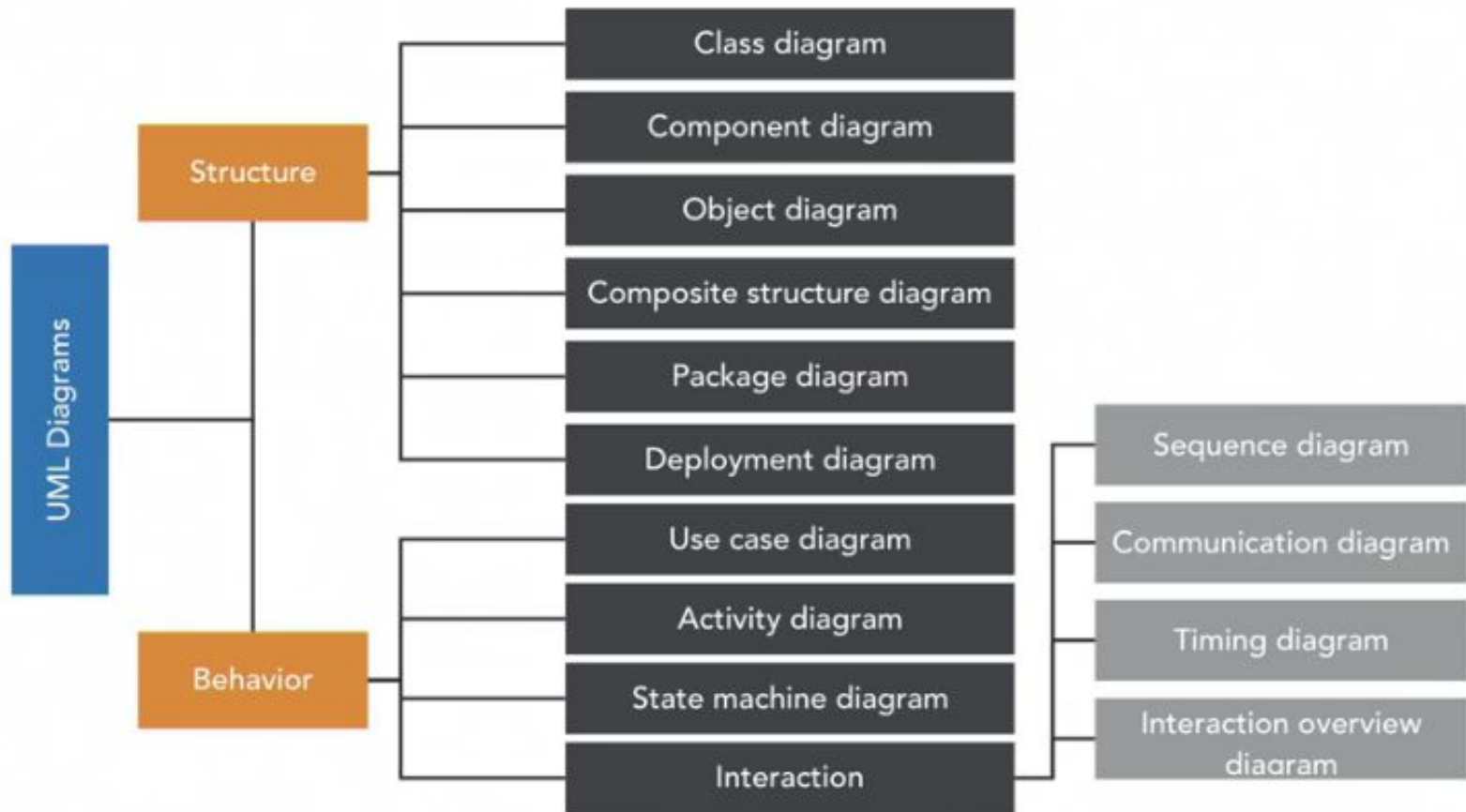
Unified Modeling Language

A family of graphical notations to describe and design software systems, especially those using an object-oriented approach

Based on standards controlled by Object Management Group (OMG)

Releases: UML 1 (1997), UML 2 (2005), UML 2.5 (2015)

<http://www.omg.org/spec/UML/>



3 Types of UML

1

Structure

Represents static view of the system and its components

2

Behavior

Represents dynamic view of the system and its components

3

Interaction

Represents interaction

- Among components of the system
- Between system and external actors

Important Considerations

- **Model selectively:** you need not (and should not) draw all the models to develop a system
- **Model collaboratively:** use models to think, share, learn, and understand together with your team
- **Model smartly:** start rough and refine it as needed, making it as a long-term asset for the team



CASE Tools

- Computer-aided software engineering tools help in various tasks throughout the software development life cycle
- Some key functions of CASE tools: modeling, code generation, reverse engineering, analyzing code complexity, and other metrics

Popular CASE Tools

Vendor	Location	Platform / OS	First public release	Latest stable release	Open source
RedSoft	China (Shenzhen)	Windows, macOS, Linux, iOS	1989	2013-03-21	Yes
Enterprise Together	Germany	Cross-platform (Java)	Unknown	2008	Yes
CaseCompass	Swiss Software	Windows	2004	2013-04	No
ConnectOne POC	CS-Systems	Windows, macOS	1993	2010-04	No
Enter Way	Enterway	Windows, Linux, macOS	2004	2015-03	No
Enterprise Architect	Sparx Systems	Windows (mostly Linux and macOS migration)	2000	2014-03-06	No
WayOne	Way One	Cross-platform (Java)	1999	2017-03-20 v1.6.0	No
Winsoft 360	Winsoft	Windows	1992	2016-04-03	No
MyUML	Germany	Windows, Linux	2003 ¹	Unknown	Yes
Provision for UML	Germania	Cross-platform (Java)	Unknown	2008	Yes
PowerDesigner	IBM	Windows	1999	2019	No
Power UML, StarUML	IBM-IT	Windows	1999	2015-05-18	No
			1999	2019-07-31 (v 1.3)	No
				2013-05-18	No
				2008-11-20	No
		Linux, Solaris	2008-08	2016-10-17	No
			2008-12	2014-10-27	No
			2011-11-12	2019-09-16	No
			2008-09-20	2019-11-28	No
			Unknown (2010)	July 2019	No
		Windows, Solaris, FreeBSD, Mac OS X, Linux	2008-09-21	2019-01-13 (v 1.0)	No
		Linux (Ubuntu)	2008-03-28 ¹	2019-03-01	No
			2008-11-18	2019-09-28	No
			1997-02	2019-09-27	No
			1998-04	2019-12-18 ¹	Yes
			Unknown	2019-08-23	Yes
			2013-08-27	2019-12	Yes
		Windows, macOS, Linux	2012	2017-04-16	Yes
		Cross-platform (Java)	2007	2009-09-19	No
		Windows, macOS, Linux, iOS	Unknown	2019-07-19 ¹	No
			2005-02-08	2016-10-23	No
			2002	2014-02-27	No
			Unknown	Unknown	No
			2002	2010-04-03	No
		Linux, Windows	2004 ¹	2012-07-26	Yes
		Windows, Linux, iOS	2008-05-18	2011-04-16	No
			2002-02	2009-11-04	No
		Cross-platform (Java)	1998-04-30	2017-09-03	Yes
		Linux, Windows	2008-05-28	2019-05-29	Yes
		Windows, macOS, Linux	2005-11-28 ¹	2019-11-24 ¹	No
		Windows	2008	2016-04-17 (v 8.0)	Yes
			2011-03-16	2017-08-14 ¹	Yes
			Unknown	Unknown	No
			Early 1990s	2019-04-16	No
			2004-10-10	2008-08	No
			2008	2019-03-18	Yes

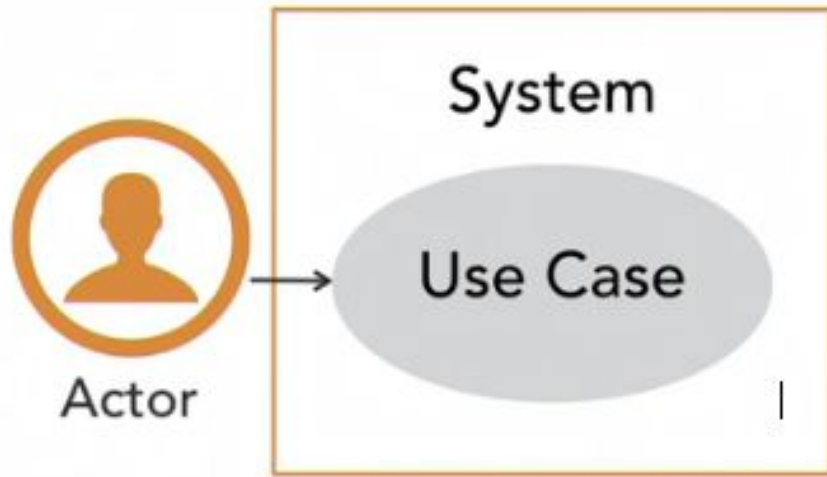
https://en.wikipedia.org/wiki/List_of_Unified_Modeling_Language_tools



Use Case Diagram

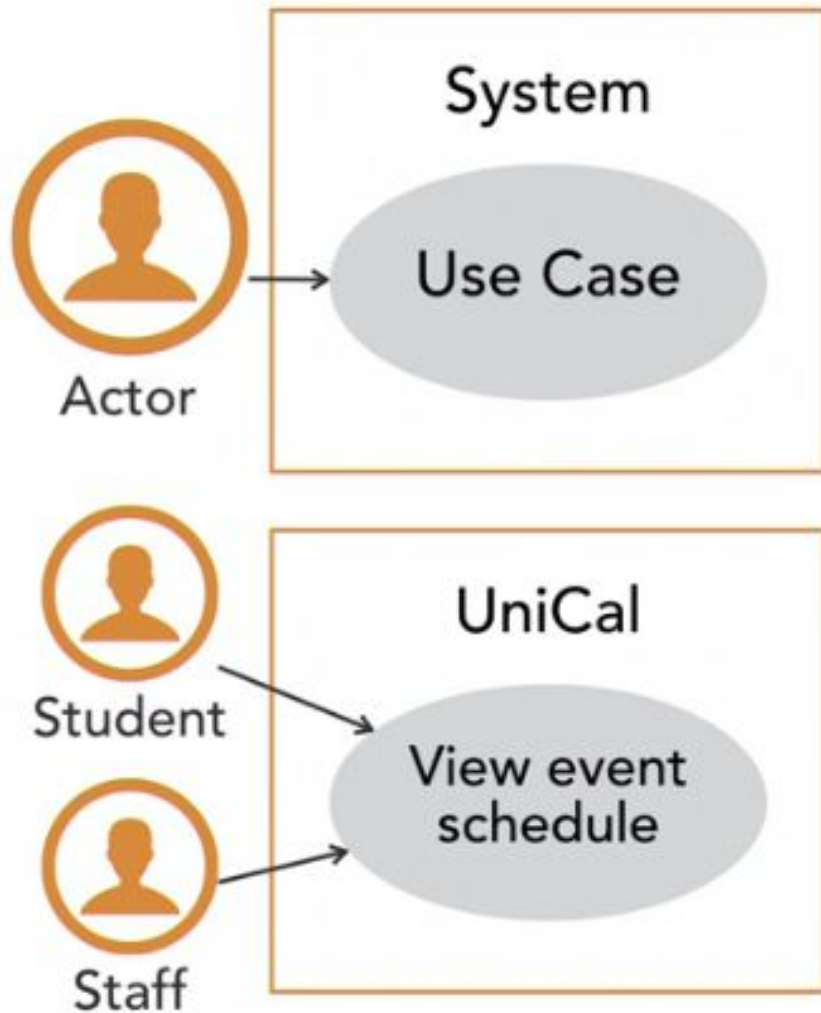
Capture high-level functionality of a system using notations for actors, use cases, and relationships among them

Often drawn by business analysts to depict the summary all use cases in a system



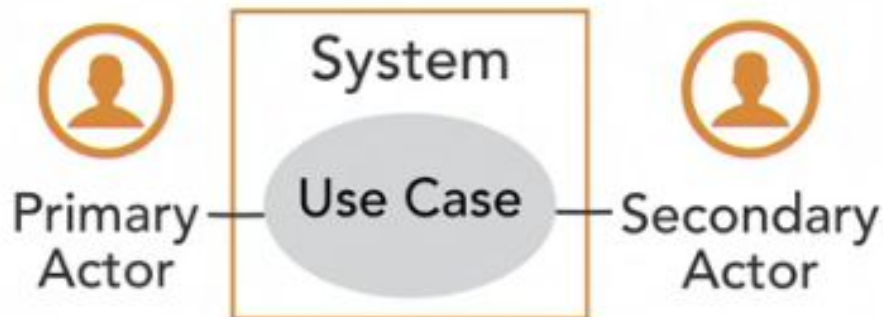
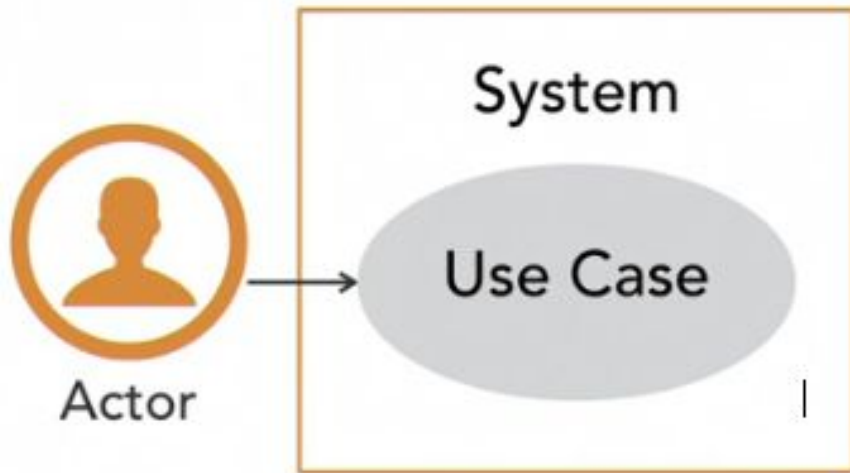
Key Elements

- Use cases
- Systems
- Actors
- Associations



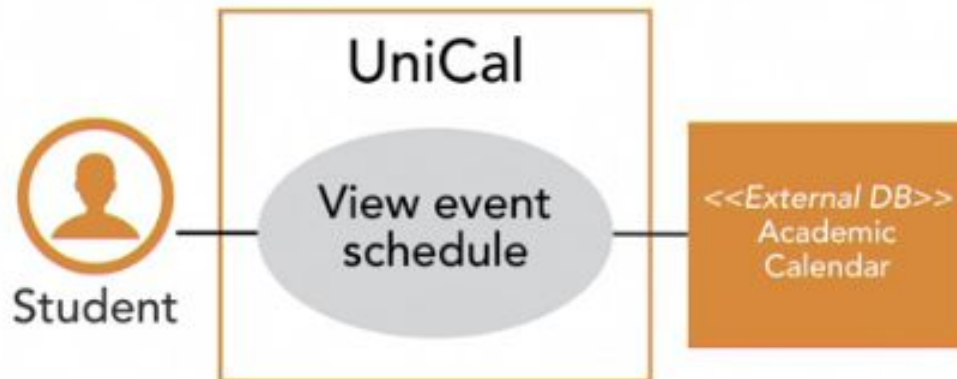
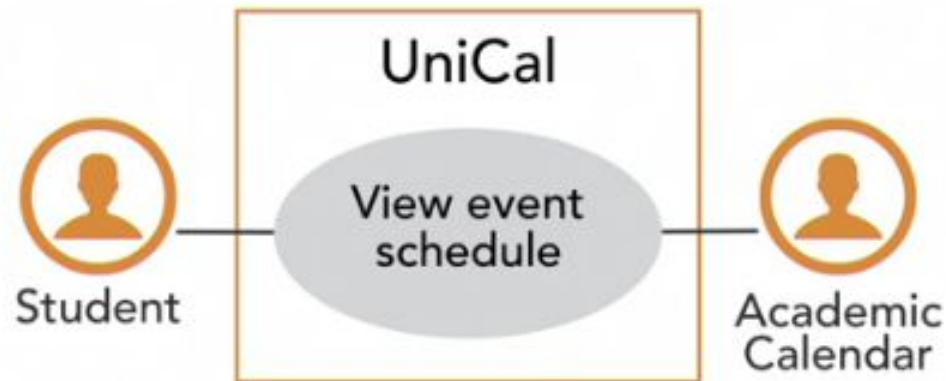
Use Case

- The use case notation is a bubble that carries use case title
- Associated with actors and possibly other user cases



Actor

- A user's role with respect to the system
- May be a human or another system
- Primary actor: whose goal is fulfilled by the use case
- Secondary actor: who is involved in the use case

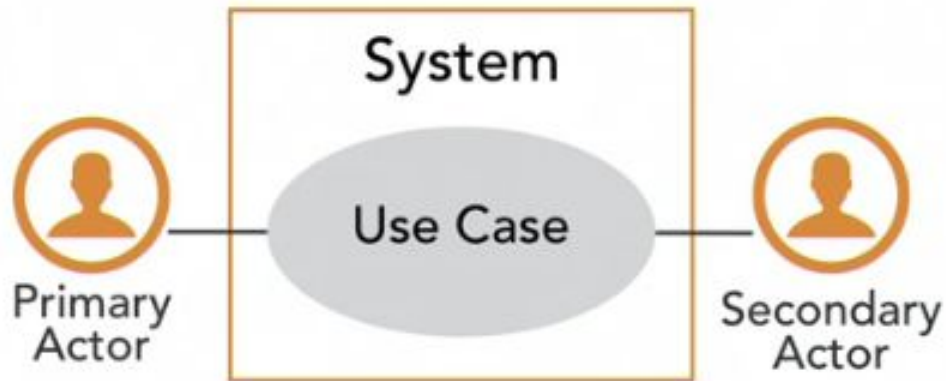


Secondary Actor

Often an external system

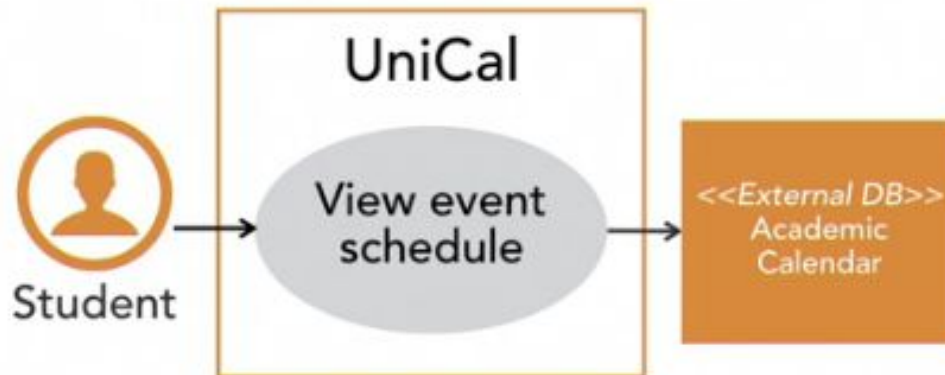
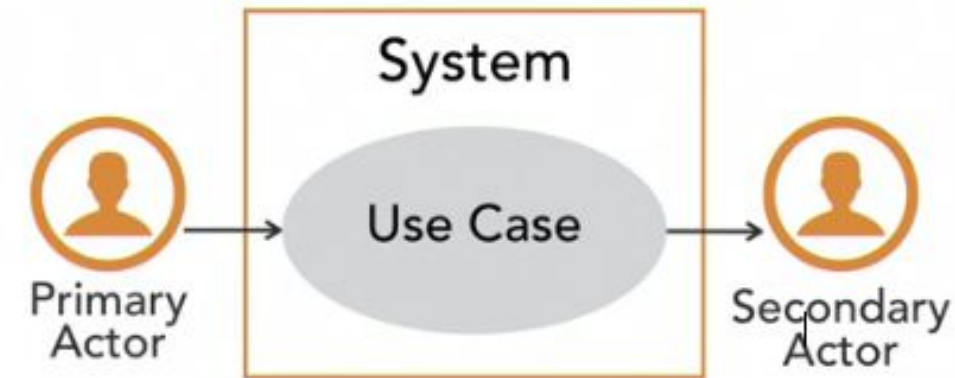
Possible notations

- A nonhuman notation such as a rectangle
- Stereotyped with `<< >>`



Association

- Between actor and use case
- Between use cases
- Between actors



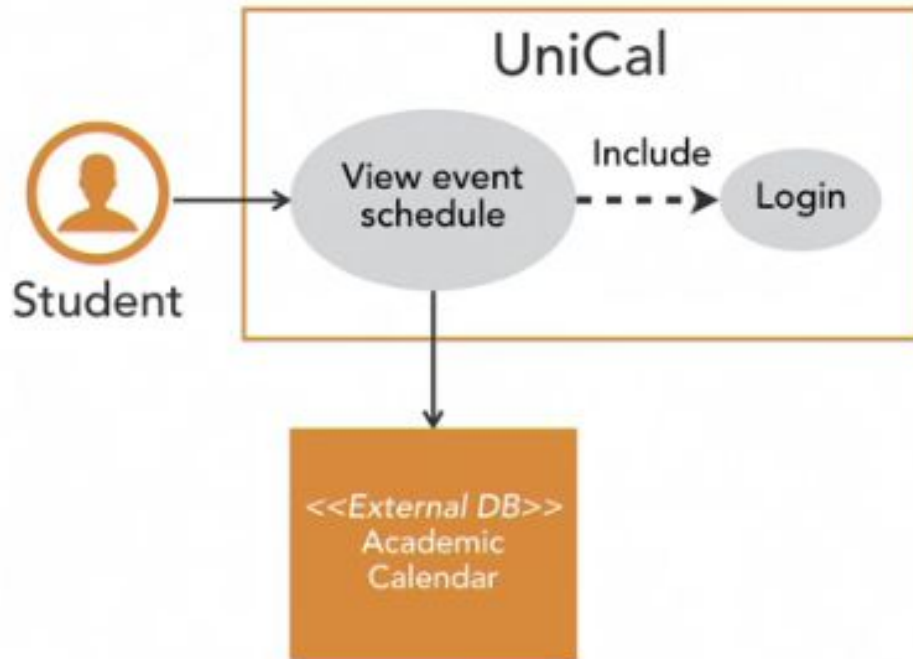
Between Actor and Use Case

Convention

- Primary actor on the left
- Secondary actor on the right of the system

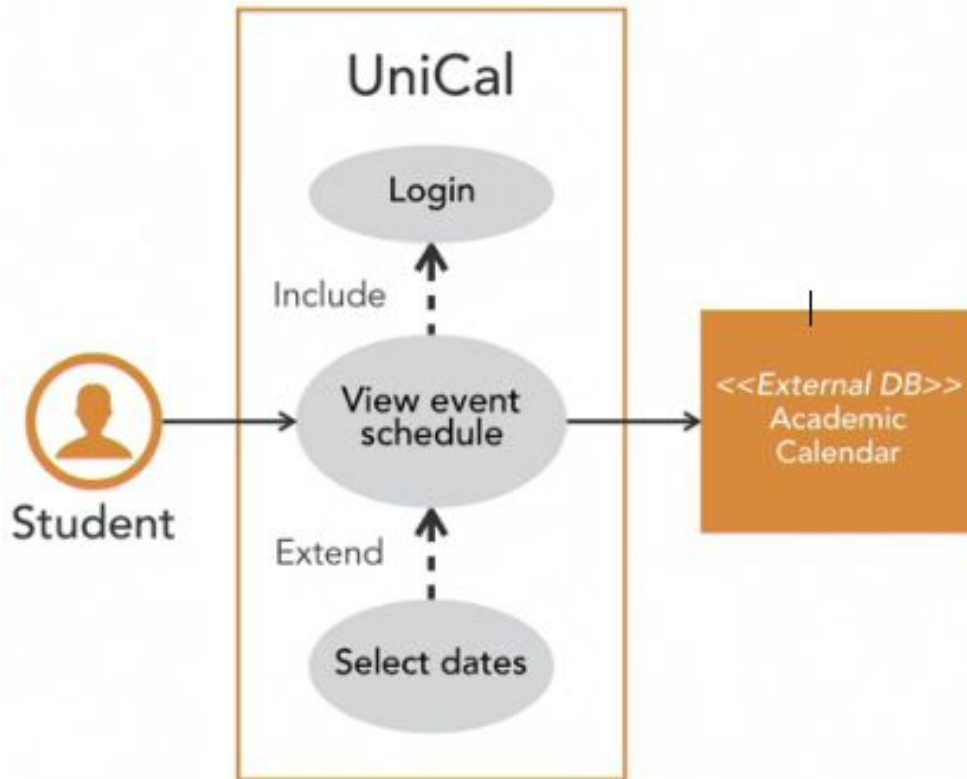
Arrowheads

- From primary to use case
- From use case to secondary



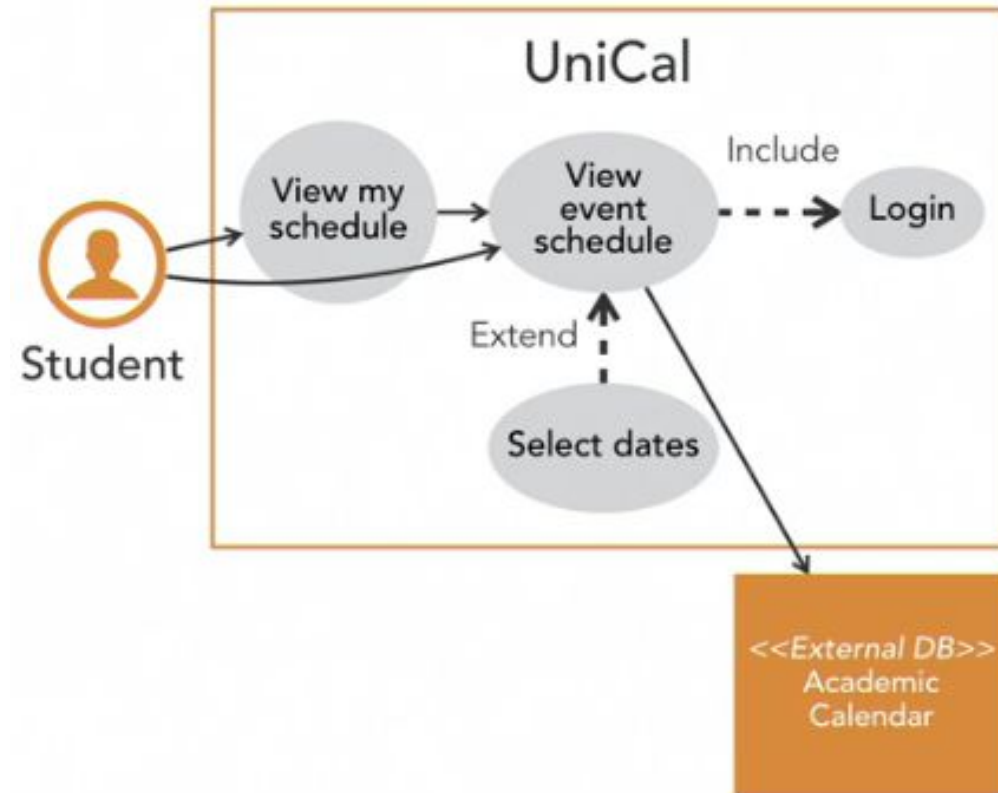
Between Use Cases: Include

- For reusable parts of behavior across two or more use cases
- Base use case depends on include use case indicated by arrow



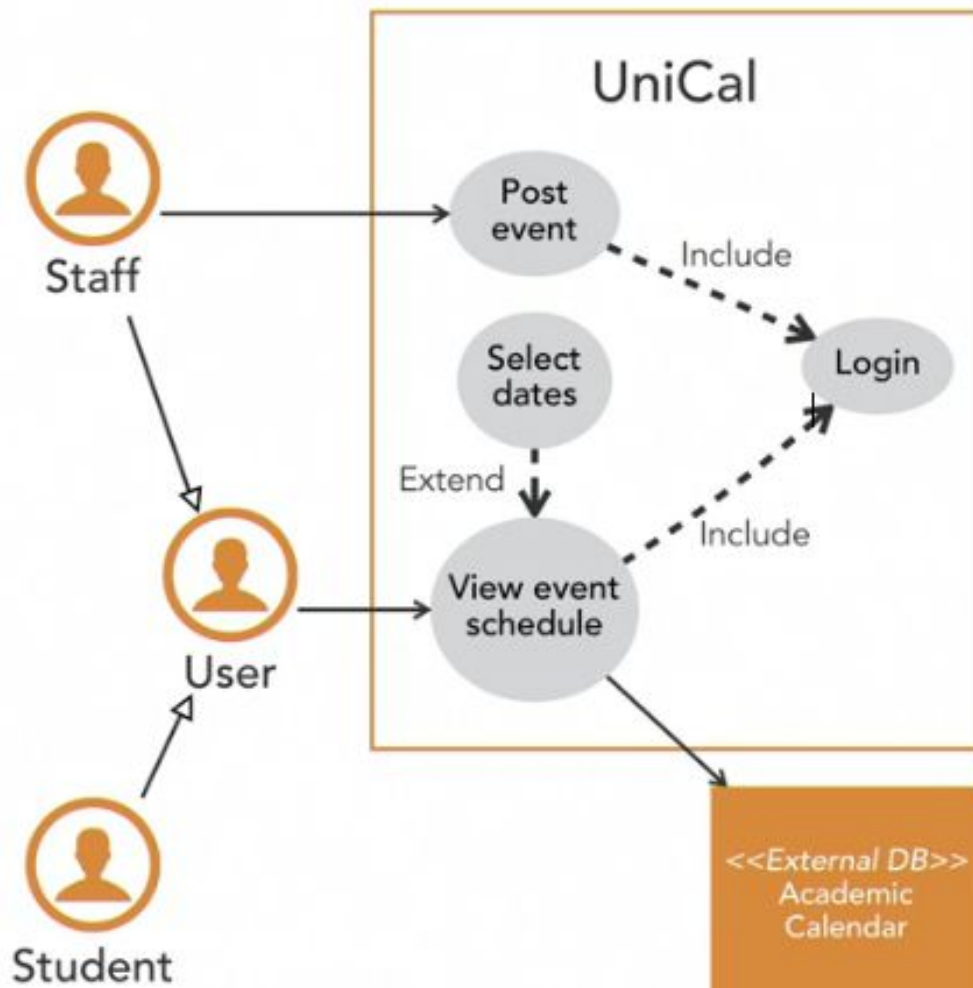
Between Use Cases: Extend

- Optional behavior added to use case
- Helps keep the base use case unchanged while adding more specifics or conditional changes
- Base use case is independent of the extend use case



Between Use Cases: Generalization

- One use case is a special case of another more general use case



Between Actors: Generalization

- Depict generalization – specialization or inheritance relationship between actors

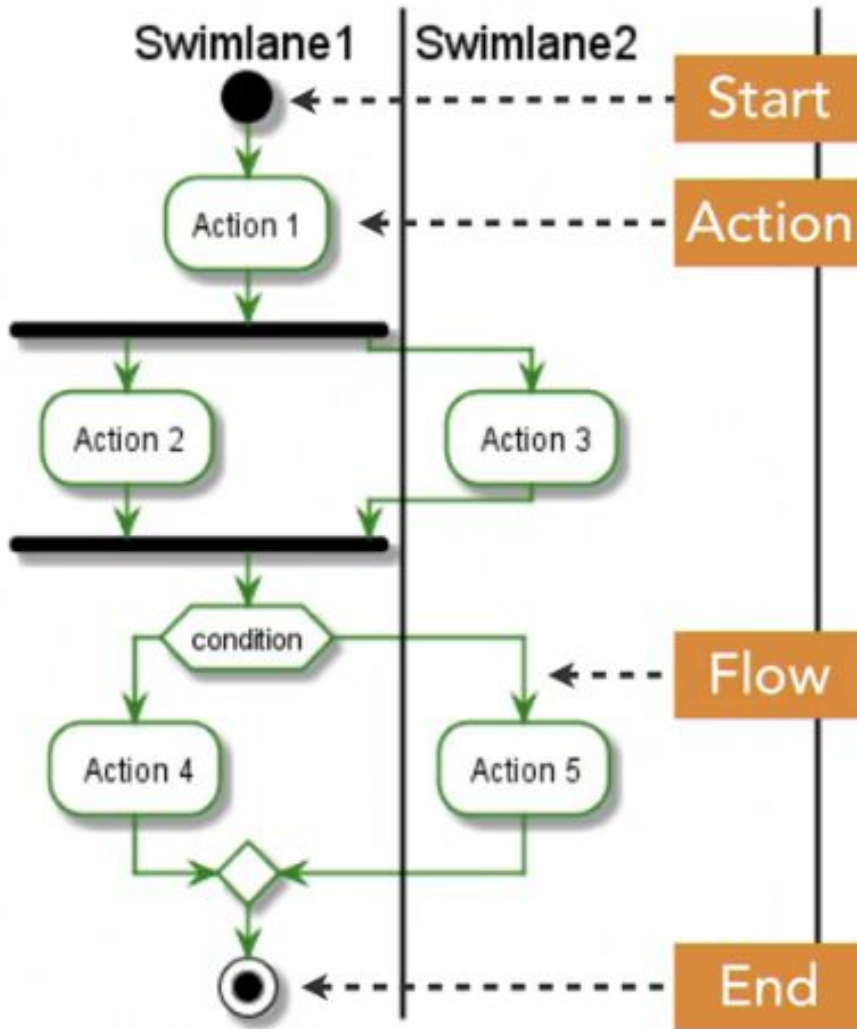


Activity Diagram

Used for workflow and process modeling

Similar to flow charts but with parallel behavior and multiple actors

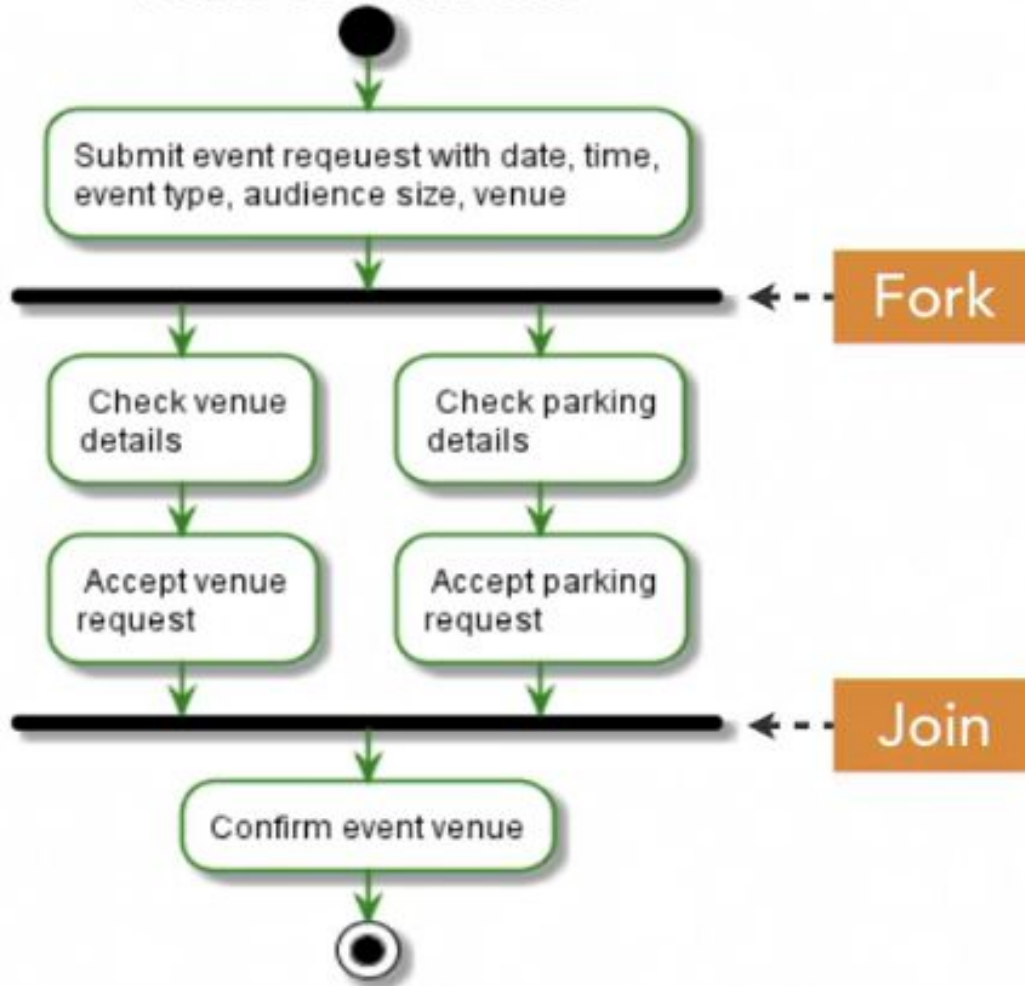
Often drawn by users, business analysts, and developers to capture their requirement understanding, for example complex scenarios in use case specifications



Key Elements

- Start and end nodes
- Actions
- Flows

Use case: Book event venue Basic scenario v1.0



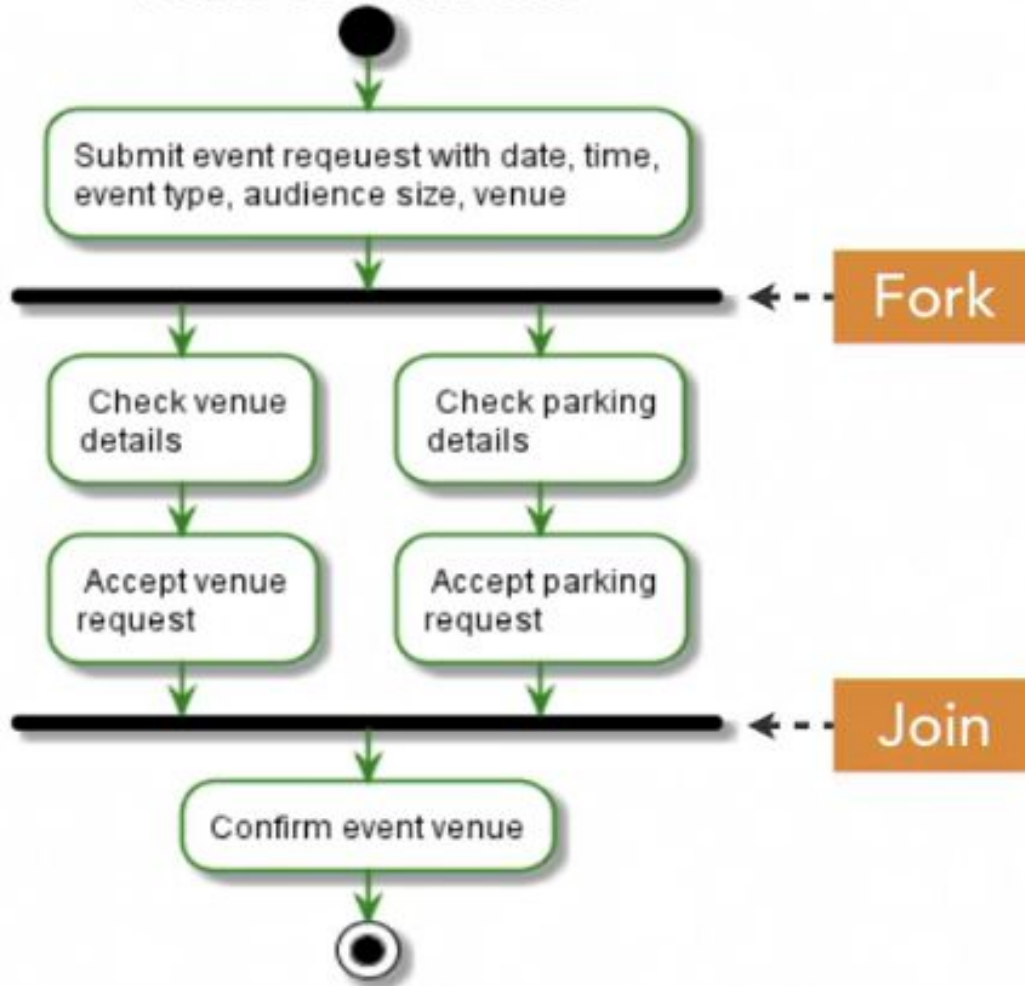
Fork and Join

Model parallel flows

Fork

- One incoming flow forks into multiple outgoing flows

Use case: Book event venue Basic scenario v1.0

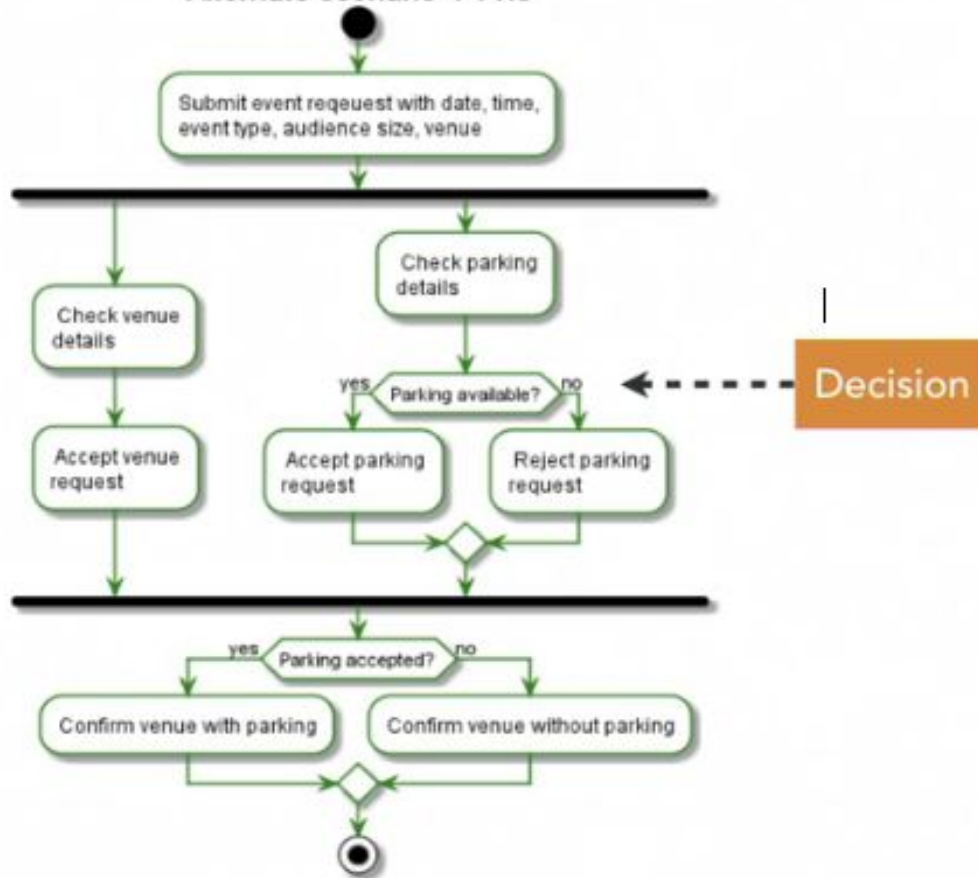


Fork and Join

Join

- Multiple incoming flows joins one outgoing flow
- Outgoing flow starts only when all incoming flows have come in

Use case: Book event venue
Alternate scenario-1 v1.0

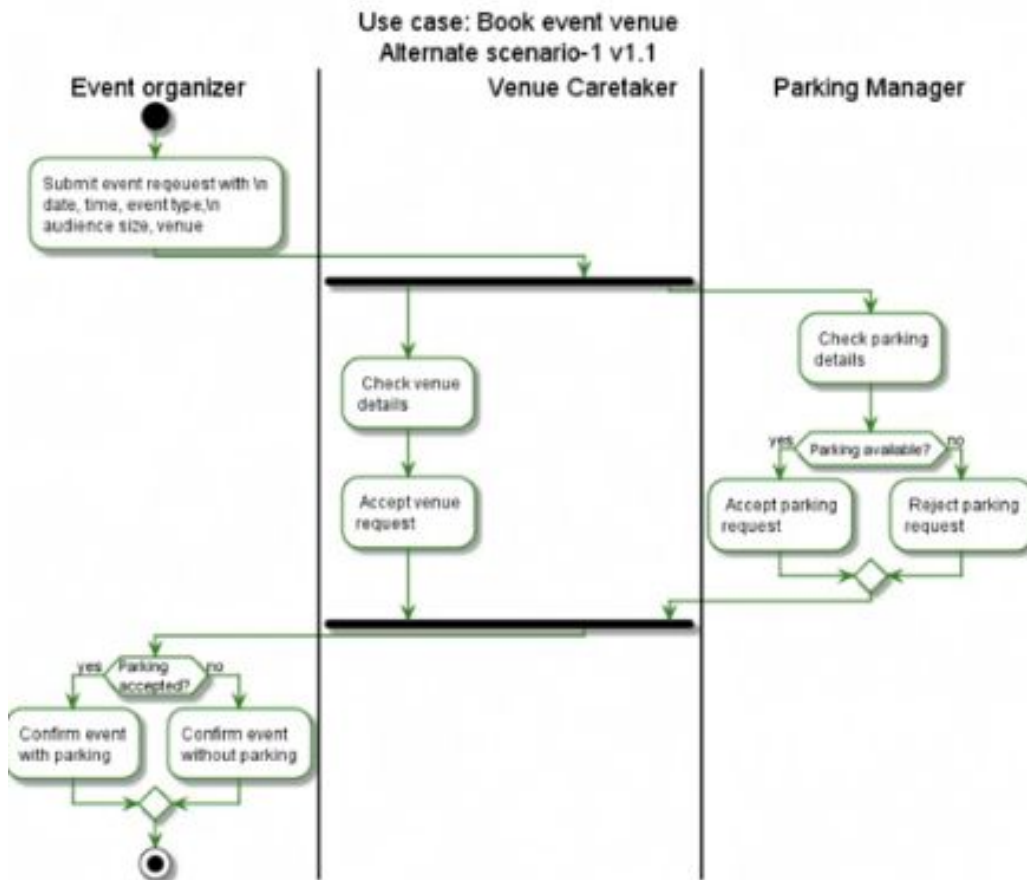


Decision and Merge

Model conditional flows

Decision

- With one inflow and multiple guarded mutually exclusive outflows
- Each outflow has a (condition) as its guard



Swimlanes

Model action “doers”

- Each doer assigned to one swimlane

Use case: Book event venue

Alternate scenario 1 v 1.2

