

Activity diagram

Overview

An Activity diagram falls under the behavioral diagramming family. An Activity can be used to model any logic - from workflow to Use Cases to methods. It focuses on flows driven by the internal processing, e.g., the implementation of an [Operation](#), or a [Use Case](#), as opposed to external events. Activity diagrams are organized according to [Actions](#).

Although Activity diagrams are often associated with Interaction diagrams, they focus more on the work performed by a system, rather than an object interaction. Additionally, they do not make the links among actions and objects very clear. Instead, Activity diagrams capture [Actions](#) and display their results.



You can create a new activity diagram under the following elements: [Action](#), [Structured Activity Node](#), [Expansion Region](#), [Conditional Node](#), [Loop Node](#), [Sequence Node](#).

Purpose

The purpose of this diagram is to focus on flows driven by the internal processing as opposed to external events. It provides a convenient way to describe complex algorithms, parallel operations, and business processes. Together with the [Communication](#) and [Sequence diagrams](#), they are used to relate [Use Cases](#). Use the Activity diagrams in situations in which all or most of the events represent the completion of internally-generated actions, i.e. procedural flow of control.

Usage

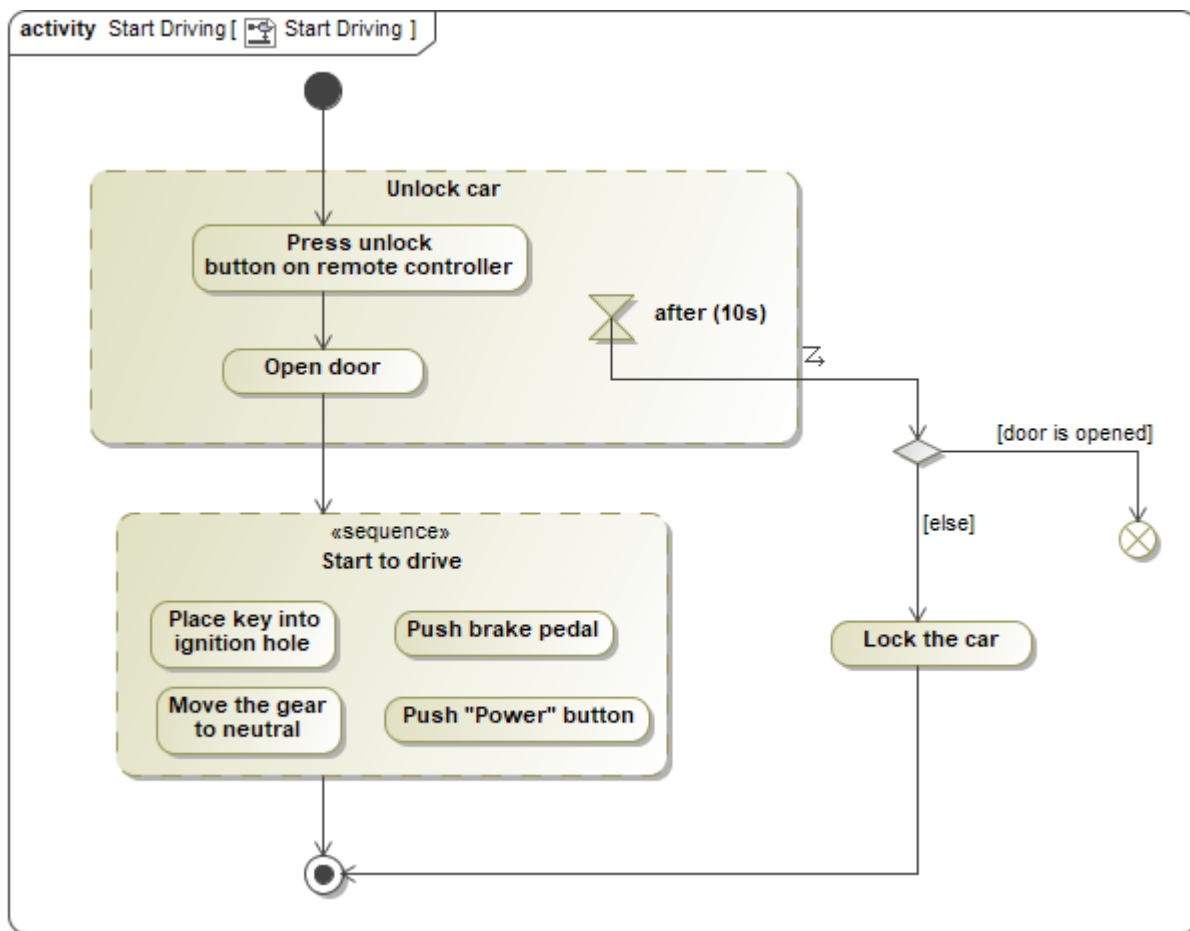
An Activity diagram can be used to:

- illustrate a business process or workflow between users and the system
- simplify and improve processes by clarifying complicated Use Cases
- understand workflow across many Use Cases
- analyze and describe the steps performed in a UML Use Case
- demonstrate the logic of an algorithm
- model method, function, operation, and other software architecture elements.

Summary

Activity diagrams are valuable because they:

- represent the logic that is necessary to implement system behaviors
- represent logic at any level the design needs, from system workflow to individual method implementations
- can be learned quickly
- are relatively familiar to users since it borrows a portion of its notation from flowcharts.
- support and encourage parallel behavior, which makes them a great tool for workflow modeling and multi-threaded programming.



Example of an Activity Diagram

Related pages

- [Creating diagrams](#)
- [Dragging in Activity diagrams](#)