effexis software

Sequence Diagram Editor
practical sequence diagram creation software

- Sequence Diagram Home
- Buy Now
- Download Free Trial
- More Information
  - Screenshots
  - Sample Diagrams
  - Documentation
- Technical Support
- Contact Us
- Press
- Resources

Only $99

Volume discounts available

Works with Windows 2000, Windows XP, and Windows 2003 Server

Comments

Great tool... nothing but praise for its ease of use and usefulness
-- Ian Taylor,
XCV Pty Ltd

I've been evaluating your software package. It's phenomenal. Having normally used Visio for sequence diagrams I'm used to spending too much time monkeying with how the diagram looks and keeping it consistent. Your sequence diagram editor gives a better graphic presentation and captures additional information.
-- Name withheld by request

UML Sequence Diagram Tutorial

What is a UML sequence diagram?

UML sequence diagrams are used to represent or model the flow of messages, events and actions between the objects or components of a system. Time is represented in the vertical direction showing the sequence of interactions of the header elements, which are displayed horizontally at the top of the diagram.

Sequence Diagrams are used primarily to design, document and validate the architecture, interfaces and logic of the system by describing the sequence of actions that need to be performed to complete a task or scenario. UML sequence diagrams are useful design tools because they provide a dynamic view of the system behavior which can be difficult to extract from static diagrams or specifications.

Although UML sequence diagrams are typically used to describe object-oriented software systems, they are also extremely useful as system engineering tools to design system architectures, in business process engineering as process flow diagrams, as message sequence charts and call flows for telecom/wireless system design, and for protocol stack design and analysis.

See also common mistakes to avoid when using sequence diagrams.

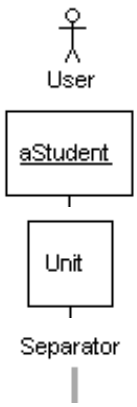
Sequence Diagram Drawing Elements

This tutorial describes the basic drawing elements used in sequence diagrams and when they are used. These are the diagram elements that are supported by the **Sequence Diagram Editor** tool. Some are not part of the UML specification and may not be supported by other UML tools.

Sequence Diagram Header Elements

The header portion of the sequence diagram represents the components or objects of the system being modeled and are laid out horizontally at the top of the diagram. See an example sequence diagram here.

Actor	Represents an external person or entity that interacts with the system
Object	Represents an object in the system or one of its components
Unit	Represents a subsystem, component, unit, or other logical entity in the system (may or may not be implemented by objects)
Separator	Represents an interface or boundary between subsystems, components or units (e.g., air interface, Internet, network)



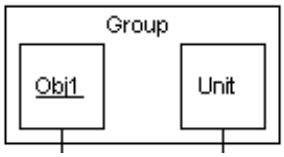
Press

Sequence Diagram Editor was mentioned in the February 1, 2005 SD Times story "Call it UML Lite." (dynamic link)

Sequence Diagram Editor was featured in the March 2005 "New and Noteworthy" column of Software Development magazine.

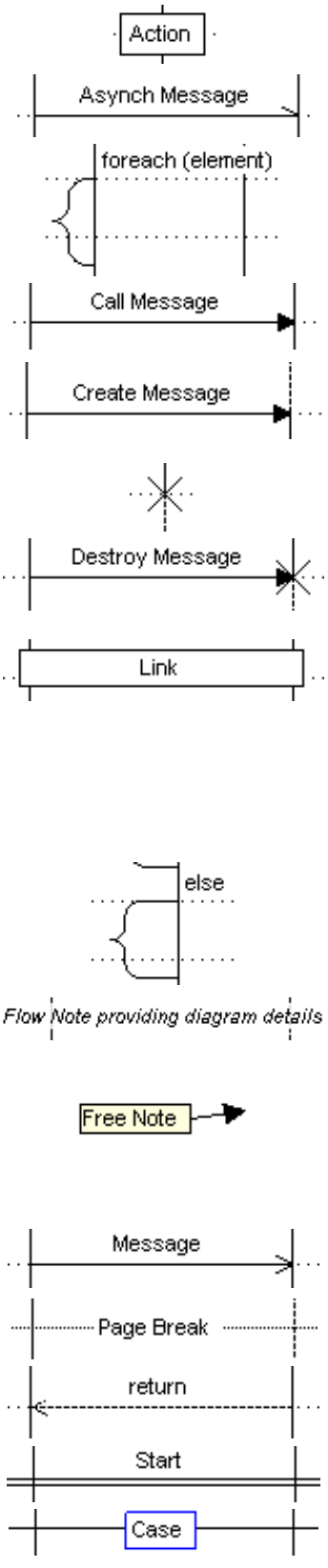
Click here for media information and press releases

Group	Groups related header elements into subsystems or components

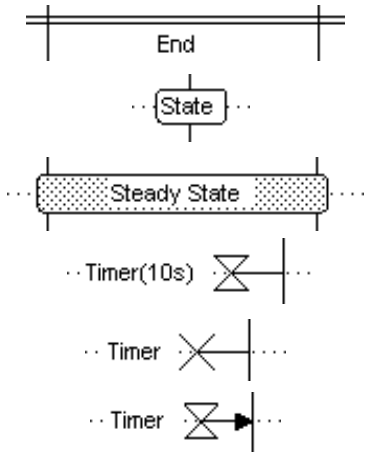


Sequence Diagram Body Elements

Action	Represents an action taken by an actor, object or unit
Asynchronous Message	An asynchronous message between header elements
Block	A block representing a loop or conditional for a particular header element
Call Message	A call (procedure) message between header elements
Create Message	A "create" message that creates a header element (represented by lifeline going from dashed to solid pattern)
Destroy Element	Represents the destruction of a header element
Destroy Message	Represents the destruction of a header element as a result of a call from another element
Diagram Link	Represents a portion of a diagram being treated as a functional block. Similar to a procedure or function call that abstracts functionality or details not shown at this level. Can optionally be linked to another diagram for elaboration.
Else Block	Represents an "else" block portion of a diagram block
Flow Note	Documentation note that is automatically formatted to flow after previous elements
Free Note	Documentation note that is free-flowing and can be placed anywhere in the diagram (can also be anchored relative to a flow element)
Message	A simple message between header elements
Page Break	A page break in the diagram
Return Message	A return message between header elements
Scenario Start	Start of a scenario (set of alternatives)
Scenario Case	Start of an alternative or case in a scenario



Scenario End	End of a scenario
State	A state change for a header element
Steady State	A steady state in the system
Timer Start	Start of a timer for a particular header element
Timer Stop	Stop of a timer for a particular header element
Timer Expiration	Expiration of a timer for a particular header element



What can be modeled using sequence diagrams?

Sequence diagrams are particularly useful for modeling:

- **Complex interactions between components.** Sequence diagrams are often used to design the interactions between components of a system that need to work together to accomplish a task. They are particularly useful when the components are being developed in parallel by different teams (typical in wireless and telephony systems) because they support the design of robust interfaces that cover multiple scenarios and special cases.
- **Use case elaboration.** Usage scenarios describe a way the system may be used by its actors. The UML sequence diagram can be used to flesh out the details of one or more use cases by illustrating visually how the system will behave in a particular scenario. The use cases along with their corresponding sequence diagrams describe the expected behavior of the system and form a strong foundation for the development of system architectures with robust interfaces.
- **Distributed & web-based systems.** When a system consists of distributed components (such as a client communicating with one or more servers over the Internet), sequence diagrams can be used to document and validate the architecture, interfaces and logic of each of these components for a set of usage scenarios.
- **Complex logic.** UML sequence diagrams are often used to model the logic of a complex feature by showing the interactions between the various objects that collaborate to implement each scenario. Modeling multiple scenarios showing different aspects of the feature helps developers take into account special cases during implementation.
- **State machines.** Telecom, wireless and embedded systems make extensive use of state machine based designs where one or more state machines communicate with each other and with external entities to perform their work. For example, each task in the protocol stack of a cellular phone goes through a series of states to perform actions such as setup a call or register with a new base station. Similarly the call processing components of a Mobile Switching Center use state machines to control the registration and transfer of calls to roaming subscribers. Sequence diagrams (or call flows as they are commonly referred to in the telecom and wireless industry) are useful for these types of applications because they can visually depict the messages being exchanged between the components and their associated state transitions.

Download Sequence Diagram Editor for a Risk Free 14-day Trial

Benefits of using UML sequence diagrams

These are some of the main benefits of using UML sequence diagrams.

1. Help you discover architectural, interface and logic problems early. Because they allow you to flesh out details before having to implement anything, sequence diagrams are useful tools to find architectural, interface and logic problems early on in the design process. You can validate your architecture, interfaces, state machine and logic by seeing how the system architecture would handle different basic scenarios and special cases.

This is particularly true for systems involving the interaction of components that are being implemented in parallel by different teams. In the cell phone example, each task would typically be implemented by a separate team. Having a set of sequence diagrams describing how the interfaces are actually used and what messages/actions are expected at different times gives each team a consistent and robust implementation plan. You can also document how special cases should be handled across the entire system.

The very act of creating the sequence diagrams and making them work with your architecture is valuable because it forces you to think about details such as interfaces, states, message order, assignment of responsibilities, timers/timeouts and special/ error cases ahead of time.

2. Collaboration tool. Sequence diagrams are valuable collaboration tools during design meetings because they allow you to discuss the design in concrete terms. You can see the interactions between entities, various proposed state transitions and alternate courses/special cases on paper as you discuss the design.

In our experience, having a concrete design proposal during design meetings greatly enhances the productivity of these meetings even if the proposed design has problems. You can narrow down the problems and then make corrections to solve them. The proposal serves as a concrete starting point for the discussion and as a place to capture proposed changes.

Sequence diagram editor makes it so easy to edit your sequence diagrams that you could even make the corrections in real time during the meeting and instantly see the result of the changes as you make them.

3. Documentation. Sequence diagrams can be used to document the dynamic view of the system design at various levels of abstraction, which is often difficult to extract from static diagrams or even the complete source code. The diagrams can abstract much of the implementation detail and provide a high level view of system behavior.

One of our colleagues shared this experience of joining the software team for a wireless switching subsystem component:

"Our component was the primary interface between the call processing component of the wireless base station and the Public Switched Telephone Network and had to support several different network protocols including T1, E1 and SS7. The previous lead had developed several dozen call flows describing the basic messaging and state transitions for common operations such as call setups including mobile and land originated, call teardowns and features such as call waiting and three way calling for each of the protocols, since each involved different states and messages with the switching component. I found these diagrams invaluable in helping me learn the software, interfaces and state machine and would refer to them often, even after working in the project for a couple of years."

See also UML Sequence Diagram Common Mistakes for things to avoid when using

sequence diagrams.

Download Sequence Diagram Editor for a Risk Free 14-day Trial

Visio UML Sequence Diagrams

Copyright © 2005 Effexis Software, LLC. All rights reserved.