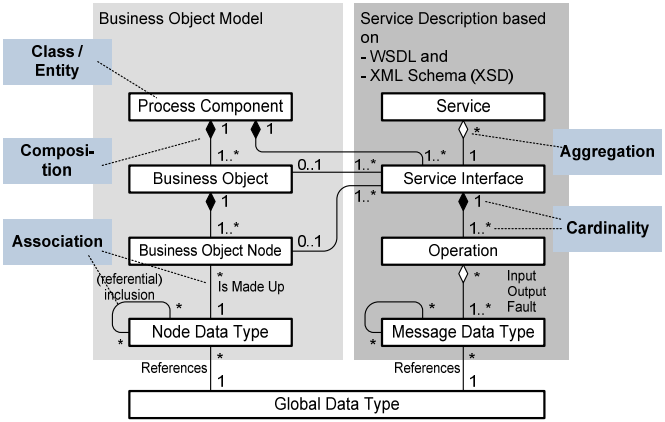
Class Diagram

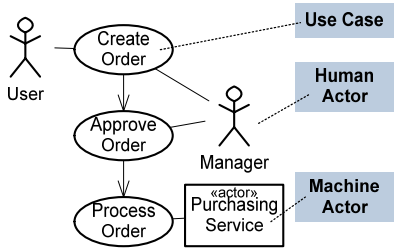
Class diagrams show the system's entities and their relationships. Use it to explain terms and their relations on high level (glossary, topic map), or to show the structure of data to be processed.



Relation to other diagram types: Use Component / Block diagrams to show where data is stored or transferred in the system.

Use Case Diagrams

Use Case diagrams allow the specification of a system's desired functionality on task level. You can show which use cases exist and who participates in them. Describe all use cases in an additional detailed text document.



Relation to other diagram types: Use Component / Block diagrams to show which agents will perform which tasks. Use Activity, Sequence or State Machine diagrams to display the behavior described by the tasks.

Design level

Component / Block Diagram

Additional components and connectors are introduced to describe the static structure of the system on design level.

Sequence Diagram

Describe method calls between objects here.

Activity Diagram

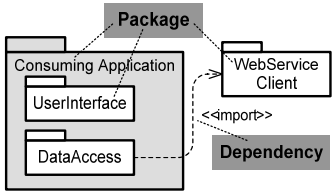
Provide more detailed information on behavior on method level.

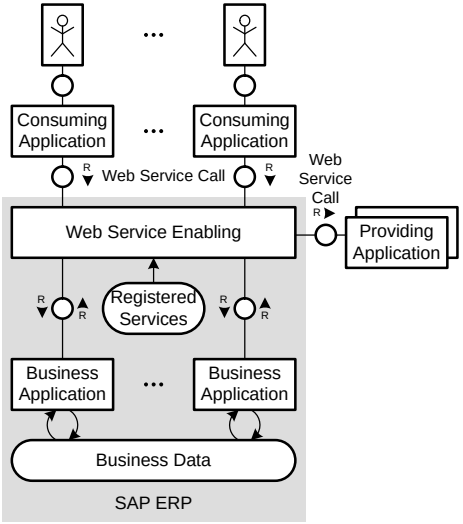
Class Diagram

Show classes and interfaces with their methods and attributes and their relations, including inheritance.

Package Diagram

Describe dependencies between packages that include classes and interfaces.





TAM™ Technical Architecture Modeling

Standard for Technical Architecture Models

What is TAM?

TAM™ stands for Technical Architecture Modeling. TAM defines a common language and a graphical notation for communication on model level at SAP. The main target is to introduce a UML standard-based set of diagram types.

- TAM can be used on conceptual and on design level
- TAM defines a set of seven UML diagram types
- TAM defines the features to be used for each diagram type
- TAM extends the UML 2.0 metamodel to incorporate FMC (www.fmc-modeling.org) block diagrams

Why TAM?

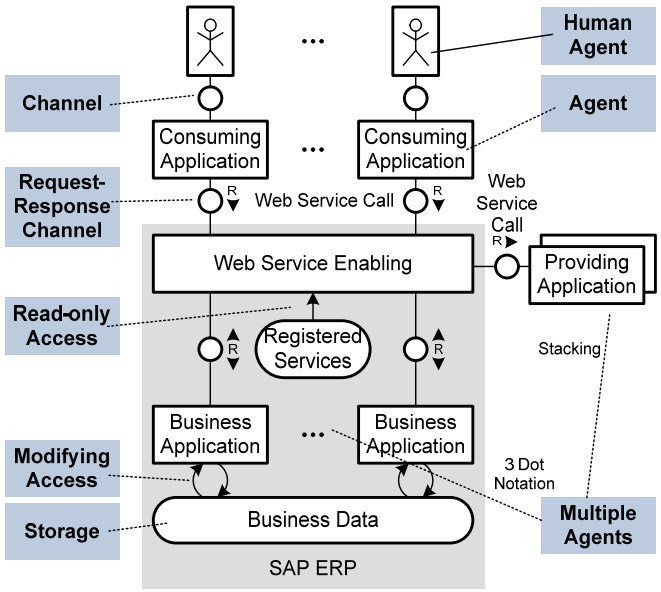
By unifying technical architecture modeling methods, you benefit in many ways:

- TAM improves the quality of architecture models by eliminating the ambiguity introduced by freestyle modeling.
- TAM facilitates knowledge exchange across teams around the world through unified architecture models.
- TAM minimizes the learning effort by reducing the number of diagram types (UML offers 13) and the variety of elements.

Conceptual Level

Component / Block Diagram

The Component / Block diagram is the most important diagram type to describe architecture. Use it to show the static structure of the system, and to provide a big picture view.



Agents perform operations; they read and write data and communicate with other agents. (Question: *Who does something?*)

Storages are passive; they only hold data that is accessed by agents. The arrows indicate direction of data flow. (Q: *Where is the data?)*

Channels are used by agents to communicate with each other. Arrows indicate the direction of data flow, the "R" indicates who initiates the request and then waits for the response. (Q: *Which agents communicate? What data and which requests do they exchange?*)

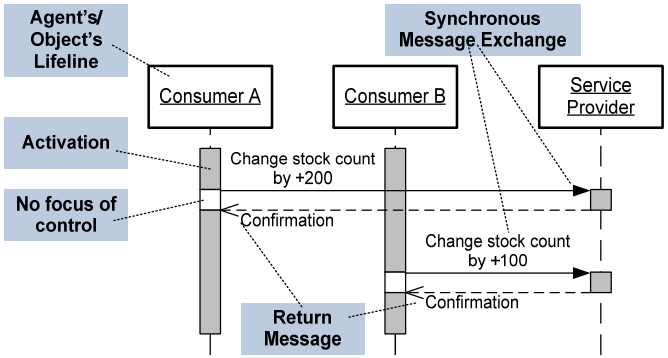
Component / Block diagrams show a snapshot view of a system on instance level; to indicate multiple agents or storages, use three dots or stack the nodes.

Relation to other diagrams types: Use Class diagrams to show how data relates that is located in storages or exchanged via channels. Use Activity or Sequence or State diagrams to express the behavior of agents and how they interact.

The Component / Block diagram is UML 2.0 compliant by an extension of the metamodel.

Sequence Diagram

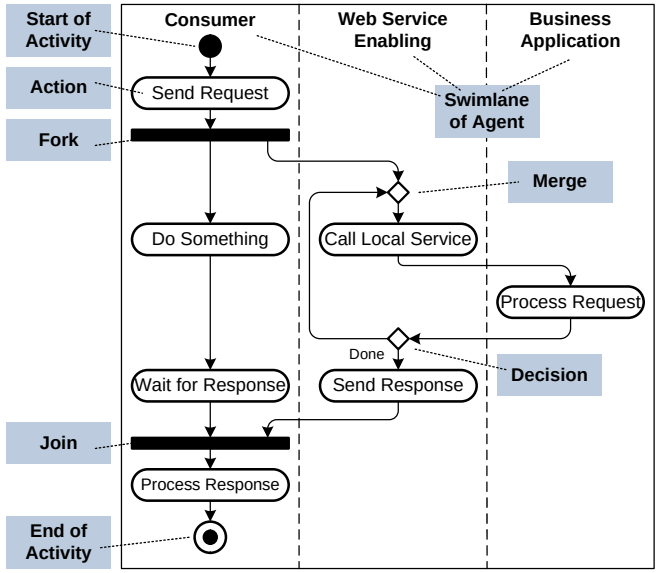
Sequence diagrams show how agents interact and communicate. They should be used on instance level, that is they should show one typical sequence.



Relation to other diagram types: The connection of the agents via channels is shown in Component / Block diagrams. To show more details of processing, use Activity diagrams.

Activity Diagram

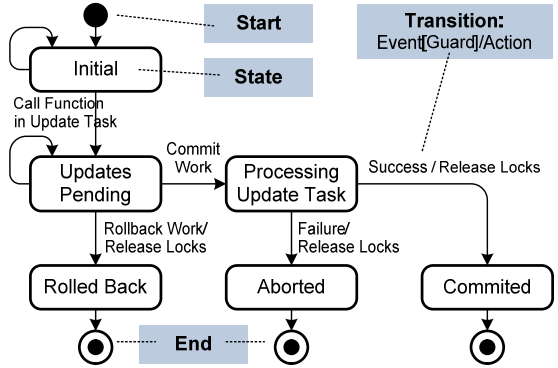
Activity diagrams show the behavior of one or more agents in greater detail. In contrast to the Sequence diagram, you can show loops or concurrency in a more intuitive way. Using swimlanes, you can assign actions to specific agents.



Relation to other diagram types: Use Component / Block diagrams to show the connections among the agents via channels, and their access to storages.

State Machine Diagram

State Machine diagrams provide an alternative way to describe behavior. Use them where you obviously need the description of states and their transitions.



A state transition is performed when the event occurs (given the guard condition is met). The transition may include an action that has to be executed as well.

Relation to other diagram types: The connection of the agents via channels is shown in Component / Block diagrams. To show more details of processing, use Activity diagrams.