UML Behavior Modeling

Behavior Modeling

- UML is essentially object oriented. Behavior exists in the context of a specific object. Behavior doesn't exist independently.
- Behavior can be described in four ways:

  Use Case UML Diagram

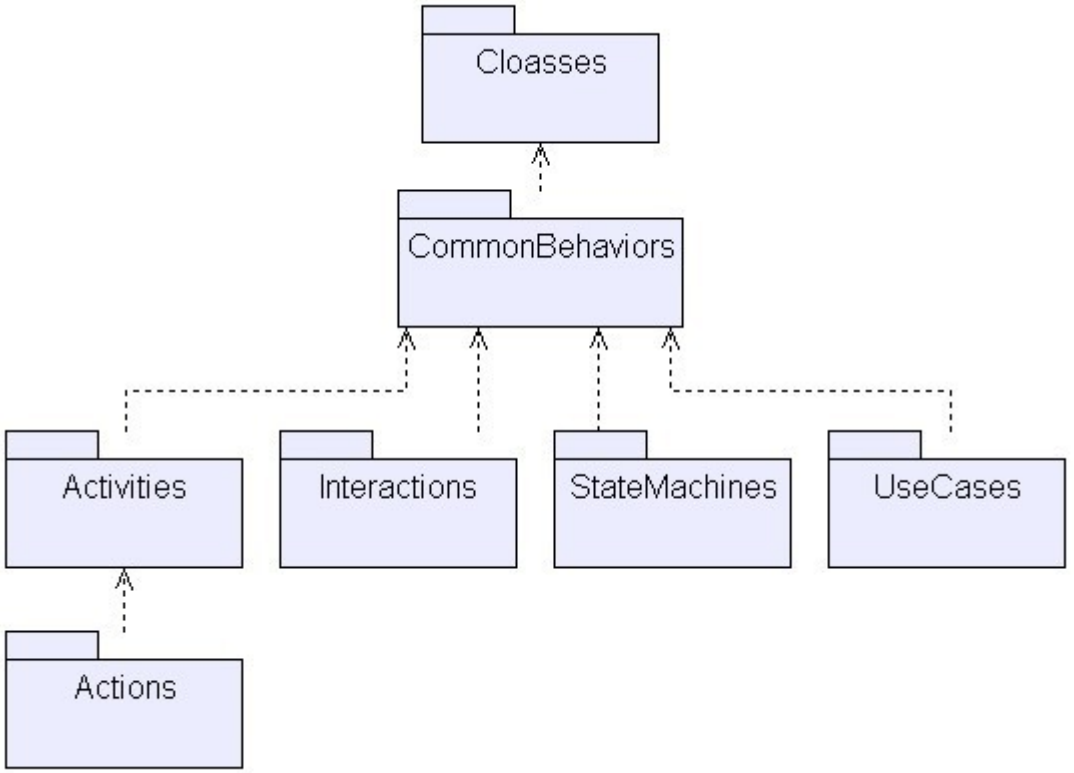
  Sequence UML Diagram

  Activities UML Diagram

  State Machine UML Diagram
- The UML meta model describes these options.

Behavior Modeling

- The UML meta model packages hierarchy for dynamic modeling:



Executing Behavior

- Executing Behavior is a behavior that is directly associated with a specific object that processes it (host) and to another object that caused this behavior (invoker).

  An airplane is an object... flying is a behavior that is directly associated with the airplane (host).The flying started because the pilot (invoker) started to fly the airplane.

Executing Behavior

- The execution of this type of behavior is triggered when the invoker calls a specific feature of the host. Calling that behavioral feature can be done either via a direct call, an event or via a new object created especially to call that behavioral feature.

  When the pilot starts flying the plane it can be involved with creating a flight plane (new created object).. When the pilot starts flying the plane it might be involved with grouping all flight personnel... guiding them.. we can treat it as an event that by initiating it the flying starts.

Emergent Behavior

- The emergent behavior refers to the interactions between different participants objects. The emergent behavior is kind of an abstraction for the behavior of several single objects... or set of objects.

  A civilian protest is an emergent behavior... those people that come to protest.. each one of them and his/her behavior... if they wouldn't have protested we wouldn't have had a civilian protest.

Describing Behavior

- A behavior can be defined as a class. Within the UML meta model we can find Behavior extending Class.
- The same way we have a Classifier responsible to create new classes we can have a BehaviorClassifier for doing the same with behaviors. Within the UML meta model we can find the BehaviorClassifier extending from Classifier.

  The BehaviorClassifier is the context of the Behavior... .the same way a Classifiers is the context of the Class. The BeahaviorClassifier can own several Behaviors. Yet, each one of them will be connected with one BehaviorClassifier only.

Calls to Behavior

- The calls to the behavior can be of several types:

  Concurrent

  Calls of this type will be executed independently of one another and concurrently with each other.

  Sequential

  There won't be any coordination between the calls. .

UML Behavior Modeling

Behavior Modeling

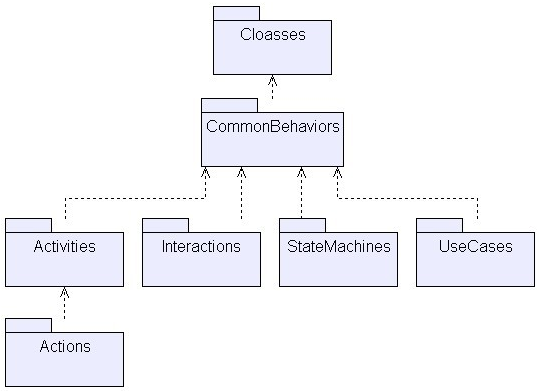
- UML is essentially object oriented. Behavior exists in the context of a specific object. Behavior doesn't exist independently.
- Behavior can be described in four ways:
  - Use Case UML Diagram
  - Sequence UML Diagram
  - Activities UML Diagram
  - State Machine UML Diagram
- The UML meta model describes these options.

The slides, starting with this one to the end of this presentation, refer to the CommonBehaviors::BasicBehaviors in the UML 2.0 specification. Unless you are already familiar with the various possible UML diagrams, it is highly recommended to skip these slides and return once you become with the various UML diagrams mentioned in these slides.

The slides, starting with this one to the end of this presentation, cover topics required for the OCUP Fundamental certification. Unless you plan to take this certification in the short run I recommend you to skip these slides.

Behavior Modeling

- The UML meta model packages hierarchy for dynamic modeling:



04/26/10 © abelski 3

The CommonBehaviors package on top includes the definitions for UML elements used to describe behavior and to describe the relationship between a behavior and a UML element.

The basic idea is that every behavior starts from an action of one (or more) objects and as a result of that the state of the participating objects changes.

Executing Behavior

- Executing Behavior is a behavior that is directly associated with a specific object that processes it (host) and to another object that caused this behavior (invoker).

  An airplane is an object... flying is a behavior that is directly associated with the airplane (host).The flying started because the pilot (invoker) started to fly the airplane.

Executing Behavior

- The execution of this type of behavior is triggered when the invoker calls a specific feature of the host. Calling that behavioral feature can be done either via a direct call, an event or via a new object created especially to call that behavioral feature.

  When the pilot starts flying the plane it can be involved with creating a flight plane (new created object).. When the pilot starts flying the plane it might be involved with grouping all flight personnel... guiding them.. we can treat it as an event that by initiating it the flying starts.

Emergent Behavior

- The emergent behavior refers to the interactions between different participants objects. The emergent behavior is kind of an abstraction for the behavior of several single objects... or set of objects.

  A civilian protest is an emergent behavior... those people that come to protest.. each one of them and his/her behavior... if they wouldn't have protested we wouldn't have had a civilian protest.

Describing Behavior

- A behavior can be defined as a class. Within the UML meta model we can find Behavior extending Class.
- The same way we have a Classifier responsible to create new classes we can have a BehaviorClassifier for doing the same with behaviors. Within the UML meta model we can find the BehaviorClassifier extending from Classifier.

  The BehaviorClassifier is the context of the Behavior... .the same way a Classifiers is the context of the Class. The BeahaviorClassifier can own several Behaviors. Yet, each one of them will be connected with one BehaviorClassifier only.

Calls to Behavior

- The calls to the behavior can be of several types:

  Concurrent

  Calls of this type will be executed independently of one another and concurrently with each other.

  Sequential

  There won't be any coordination between the calls. .

