UML 2 for Systems Engineering

Conrad Bock
NIST

conrad.bock@nist.gov
Unified Modeling Language

- Originated in object-oriented software community.
- However:
  - Wide lifecycle, including logical specifications and deployment.
  - More than pictures:
    - Includes a repository model/API and
    - … and XML interchange.
  - Behavior models with virtual machines.
  - Not just for software modeling.
Wide Lifecycle

- Logical and physical modeling
  - Logical entities
  - Physical and software entities
    - Environment
    - System
- Requirements
  - Use cases
  - Constraints
- Deployment
  - Artifacts
  - … tied to specification and delivery.
More Than Pictures

- Repository provides
  - API’s
  - XML interchange
  - Support for multiple notations

- UML notation stores to repository
  ... and alternate notations can, too.

- Generate systems from repository:
  Notation → Repository → System
Model-Centered Development

Class Car
{ color : Color [1..*];
  driver : Person [1..1]
}

Class Person
{ car : Car [1..*];
}

Presentation

Parsing

Repository

Model compilation

Actual system

Class Car
{ color : Color [1..*];
  driver : Person [1..1]
}

Class Person
{ car : Car [1..*];
}
Behavior Models

- Multiple types of model.
- None dominant.

Different emphasis in each one:
  - Activity models
    - Series of actions
  - Interaction models
    - Messages between objects
  - State machines
    - Objects reacting to events

- Virtual machines defined for execution.
UML Status

- First version adopted in 1997 (1.1).
- Minor revision in 1999 (1.3).
- Coding models added in 2001 (1.5).
- Major revision finishing now (2.0).
- Primary submission is created by many modeling vendors and users.
New Areas in UML 2 for SE

- Activity model supports physical as well as computational processes.
- Composition model supports inter-part connections.
- Deployment model ties specification to delivered system.
- Information Flow.
- Time model.
- And others.
Activity Modeling

- Activity modeling emphasizes the sequence and conditions for coordinating other behaviors
- ... using secondary constructs to show which classifiers are responsible for those behaviors.

- Focus is on what tasks need to be done, in what order, rather than who/what performs each task.
Activity Modeling

- Tasks and ordering ...

```
Receive Order -> Fill Order [order accepted] -> Ship Order -> Close Order
   |                                |
   | Send Invoice                  |
   | Invoce                        |
   | Make Payment                  |
   | Accept Payment                |
```
Activity Modeling

- ... plus resource assignments.
UML 2 Activities for SE

- **First-class behavior model:**
  - Usable with or without objects
  - Parameterized
  - Behavior properties

- **Full parallelism**
  - Concurrent branches operate independently.

- **Input/output**
  - Queuing, storage
  - Notation
  - Multi-entry/exit

- **Full action model**
  - For model execution and simulation.
First-class Behavior Model

- Object-orientation not required to model dynamics ...
- ... but supported when needed.
- Behaviors can be invoked directly, or through an object owning the behavior.
- Parameterized for input/output.
- Can have attributes, associations, operations, states, ...
First-class Behavior Model

POEmployee
- sortMail()
- deliverMail()

Deliver Mail
- Keys
- Check Out Truck
- Put Mail In Boxes

Deliver Mail
- HowLong : Time
- Abort()

Truck

resource
0..1
1
State-based UML 1.x Activities

(Assignment)

(Activity)

(State Machine)

Trace: A, B||X, C||Y, Z
Token-based UML 2 Activities

(Activity)

Diagram:

```
A  --(Token flow, not a notation)-- B  --C-- Z
     \                  /            \
     X                  Y             X
```

Trace: A, (B,C), Z || (X,Y)
Unrestricted Parallelism in UML 2

(Activity, tokens not notation)

Trace: \( A, (B,C), Z \parallel (X,Y) \parallel N \)
Queuing

- Tokens can
  - stack up in “in/out” boxes
  - backup in network
  - prevent upstream behaviors from taking new inputs

- Applicable to systems with significant resource constraints, such as physical or manual processes.
Queuing

- Tokens can be
  - Stored temporarily
  - Divided between flows
- Tokens cannot
  - Flow in more than one direction, unless copied.
Non-queuing

- No token interaction.
- For domains without resource constraint, such as computation.

```c
Amount function update_account
    (a : Account, d : Amount)
{
    Amount nb = a.balance + d;
    a.balance = nb;
    send_notice (a.customer, a, nb);
    return nb;
}
```
Non-queuing

update_account

Deposit → + → Amount
Account → Get Balance
Get Customer
Get Balance → Set Balance
Set Balance → Send Notice
Send Notice
Parameter Sets

- Sets of parameters can take input or provide output ...
- ... exclusive of each other at runtime.
- See multi-exit in EFFBD.
Full Action Model

- Actions are the “steps” in an activity (round-cornered rectangles).
- Include:
  - Invoking behaviors/functions.
  - Creating/destroying objects.
  - Getting/setting property values.
- For fully-executable models and simulations.
Extended Functional Flow Block Diagram

- Control/data flow diagram.

Extended Functional Flow Block Diagram

- Most of EFFBD supported by UML 2 Activity diagrams.
- Some differences in execution …
- … to be addressed in SE profile for UML 2 or in minor revision to UML.
Function ↔ Behavior/Action

- EFFBD Function and UML 2
  Action/Behaviors are steps in a process flow.

  (EFFBD)
  | # |
  | Move Elevator |

  (UML 2)
  Move Elevator

- Notation is different, but repository would be the same (except for adding #).
Control Flow

- EFFBD and UML 2 Control Flow give time sequence to steps in a process flow.

(EFFBD)

(UML 2)
Data/Object Flow

- EFFBD and UML 2 Data Flow specify how Function/Behavior outputs are provided to inputs.

(EFFBD)

(UML 2)
External I/O ↔ Parameter

- EFFBD External Input/Output and UML 2 Parameter support I/O at the beginning/end of the entire diagram.
Select $\leftrightarrow$ Decision

- EFFBD Select and UML 2 Decision specify mutually exclusive paths in a flow.

(EFFBD)

(UML 2)
Concurrency ↔ Fork/Join

- EFFBD Concurrency and UML 2
- Fork/Join specify parallel paths
Multi-exit $\leftrightarrow$ Parameter Sets

- EFFBD multi-exit functions and UML 2 Parameter Sets specify mutually exclusive outputs.

(EFFBD)

completion condition

(UML 2)

TBD: Postcondition on parameter set
Cycles

- EFFBD and UML 2 flows can have cycles in the flow graph.
Edge Shortcuts

- Notational shorthand for long flow lines:

![Diagram of edge shortcuts]

is equivalent to

![Diagram of edge shortcuts (equivalent)]
Example EFFBD

Adapted from Long, James, "Relationships between Common Graphical Representations in System Engineering", ViTech Corporation, www.vitechcorp.com
UML 2 Translation

1. Serial Function
   - Data 1

2. Multi-exit Function
   - Data 2

3. Function in Concurrency

4. Function in Multi-exit Construct

[ before third time ]

5. Function in an Iterate
   - [ else ]
   - Data 3
   - Data 5

6. Output Function
   - Data 4
To Be Addressed

- Triggering and non-triggering inputs.
  - Distinguish required from streaming.
- Multi-exit, queuing on control.
  - Treat control as a kind of data.
- Completion conditions on multi-exit.
  - Add postconditions to UML 2 ParameterSet.
- Control/data flows that disable functions.
  - Extend control tokens.
- Iteration
  - Extend merge node.
- Continuous data flows.
  - Fine-grained token flow.
EFFBD ~ Activities

- Significant similarity between EFFBD and UML 2 Activities.
- Entry point for SE’s into UML.
- Integrates with other UML features useful to SE:
  - Classes
  - Composition (Assembly)
  - Information Flow
  - Many other features not presented here.
- Details of EFFBD ↔ Activity translation at:
  http://www.u2-partners.org/outgoing/syseng/seu2pactivitymap.zip
Composition UML 1.x

- UML 1.x supported part-whole ...
Composition UML 1.x

... but not part-part:

- Power to wheels on other people’s cars
- Propellers on cars
- Wheels on boats
- Associations are global

(not legal UML anyway)
Composition UML 2.0

- UML 2 supports part-part in context:

![UML Diagram]

- Engine as used in Car
- Engine as used in Boat
- Powers as used in Car
- Powers as used in Boat
Composition UML 2.0

- Enables better abstraction:

  ![UML Diagram]

  Global structure inherited by each kind of Vehicle …
  … and constrained for each kind
Interfaces UML 1.x

- UML 1.x supports interfaces, but only in one direction:

- Interface usage buried in client methods.
Interfaces UML 2.0

- Bidirectional interfaces:

- **Car**
  - Engine
    - PowerOut
  - : Wheel
    - PowerIn

- **Boat**
  - Engine
    - PowerOut
  - : Propeller
    - PowerIn
Composition 2.0 (Ports)

- Ports = public parts.

- Car:
  - Engine: PowerOut
  - Wheel: PowerIn

- Boat:
  - Engine: PowerOut
  - Propeller: PowerIn
Composition 2.0 (Ports)

- Multiple ports of the same type.

```
<table>
<thead>
<tr>
<th>Home Office</th>
</tr>
</thead>
<tbody>
<tr>
<td>: PowerStrip</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>PowerOut</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>: Printer</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>PowerOut</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>: Lamp</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>PowerOut</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
```
Information Flow

- Very abstract flow model.

- Independent of message passing, parameters, etc, but can be tied to these.

- Applicable to Elaborated Context Diagrams
Time Model

- Can be used to state constraints on processes:

```
<table>
<thead>
<tr>
<th>Close Doors</th>
<th>Move Elevator</th>
</tr>
</thead>
</table>
```

< 2 sec
Interaction Model

Gate

Interaction Operator

Operand separator

s décoexample

Gate

[x>0]  [else]  [else]

create()

bar(x)()

more()

foo(x)()

doit(z)()

doit(w)()

opti()
Messages can overlap in time.
Deployment

- Support for general mappings between design elements, artifacts, and deployment targets.
- Communication paths between nodes.
  - Supports locality diagrams.
Summary

- UML is
  - wide lifecycle
  - applicable to multiple domains
  - a repository for multiple notations
- UML 2 adds new models useful to systems engineering ...
- ... in both structure and behavior.
- Latest draft of UML 2 submission: http://doc.omg.org/ad/03-03-02.
- Updates: http://www.omg.org/uml