4D Composite Structure: (Onto) Logical 4D Modeling

Conrad Bock
U.S. National Institute of Standards and Technology

Charles Galey
Lockheed
Overview

- RoadMap
- Motivation
  - Behavior, review
  - 4D, requirements
- 4D Solution
  Objects and behaviors
  1. in space-time
  2. within time intervals, space regions
- Summary
Overview

- RoadMap
- Motivation
  - Behavior, review
  - 4D, requirements
- 4D Solution
  Objects and behaviors
  1. in space-time
  2. within time intervals, space regions
- Summary
# Behavior as Composite Structure

<table>
<thead>
<tr>
<th>Onto 4D</th>
</tr>
</thead>
<tbody>
<tr>
<td>(this one)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Onto Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>(ad/19-06-02)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Onto State Machines, Parts 1 &amp; 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>(ad/18-12-09, 19-03-02)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Onto Interactions</th>
<th>Onto OO</th>
</tr>
</thead>
<tbody>
<tr>
<td>(ad/18-06-11)</td>
<td>(ad/18-09-07)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Onto Behavior Basics</th>
</tr>
</thead>
<tbody>
<tr>
<td>(ad/2018-03-02)</td>
</tr>
</tbody>
</table>
Overview

- RoadMap
- Motivation
  - Behavior, review
  - 4D, requirements
- 4D Solution
  Objects and behaviors
  1. in space-time
  2. within time intervals, space regions
- Summary
Original Problem

- UML has **three behavior diagrams**.
  - Activity, state, interaction.
- **Very little integration or reuse** between them.
  - Three underlying metamodels.
  - Three representations of temporal order.
- **Triples the effort** of learning UML and building analysis tools for it.
General Solution

- Treat **behaviors as assemblies of other behaviors**.
  - Like objects are assemblies of other objects.

- **Assembly = UML internal structure**
  - Pieces represented by **properties**.
  - Put together by **connectors**.

- Put all behavior diagrams on the same underlying behavior assembly model.
Behaviors as Composite Structure

- **Property**
- **Connector**

**Tire Traction**

**State Machine**

**Activity**

**Sequence Diagram**

**Prevent Lockup**

**Interaction**
Behaviors model “things” happening over time.

- With temporal relations (time constraints) between them.
The TakePicture occurrence on the right does not follow the behavior model.
Behavior as “Composite Timing”

- Composite structure relations are temporal:
  - Part-whole = happens during.
  - Part-part = happens before.

**Model (M1)**

**Things Being Modeled (M0)**

**Behavior**
- Focus
- Shoot
- Take Picture

**Time**
Behavior as “Composite Timing”

Model (M1)

Property (whole-part)

Connector (part-part)

class TakePicture

step1: Focus

: HappensBefore

step2 : Shoot

Focus

HappensBefore

Shoot

Not instance specs

Things Being Modeled (M0)

TakingPic1:

step1

:Focusing DuringTP1:

Happens Before

step2

Shooting DuringTP1:

TakingPic2:

step1

:Focusing DuringTP2:

Happens Before

step2

Shooting DuringTP2:

Focusing before shooting in same taking picture
Dashed arrows between M1 and M0 mean ....
M0 $\rightarrow$ M1 Synonyms

Classified by
Modeled by
Specified by
Conforms to
Follows
Satisfies (logically)

Not quite: Instance of (in the OO sense)
Not at all: Execution of (in the software sense)
Behavior: What’s Being Modeled?

“Things” that occur in time
- Eg, taking a picture, focusing, etc.
- Not “behaviors”, “actions”, etc.

Real, Simulated, or Desired Things Being Modeled (M0)

- Not instance specs.

Focus
3/15/09 10-11pm ET:

TakePicture
3/15/09 10-12pm ET:

Shoot
3/15/09 11-12pm ET:
**Behavior: What’s in Common?**

- They happen before or during each other.
  - Construct M1 library for this.
  - Use it to classify things being modeled.
Behavior: Use Library

- Specialize library classes and subset/redefine library properties.
Behavior: Too repetitive at M1?

- Capture M1 patterns in M2 elements.
  - Tools apply patterns automatically.
Benefits: Original Problem

- Flexibility in using metamodels
  - Add metaelements as needed to simplify library usage.
- Many metaelements become synonyms
  - Application / method / diagram-specific terminology sharing same semantics.
  - M2 actions, states, etc, => M1 happensDuring
- Learning UML and building analysis tools for it is easier
  - Due to shared semantics for variety of modeling language terminology.
Benefits: Expressiveness

- Constraints are inherited in UML
  - including temporal constraints.
Benefits: Expressiveness

- Combine activity and state machines.
  - States and actions happen during their “containing” occurrences, ordered in time.
Benefits: Modeled Semantics

- **UML semantics is written in free text**
  - Specifying an execution procedure for activities and state machines:
    - Tokens are offered to an ActivityEdge by the source ActivityNode of the edge. Offers propagate through ActivityEdges and ControlNodes, according to the rules associated with ActivityEdges (see below) and each kind of ControlNode (see sub clause 15.3) until they reach an ObjectNode (for object tokens) or an ExecutableNode (for control tokens and some object tokens as specified by modelers, see ObjectNodes in sub clause 15.4). Each kind of ObjectNode (see sub clause 15.4) accepts Activity which a
  - The processing of Event occurrences by a StateMachine execution conforms to the general semantics defined in Clause 13. Upon creation, a StateMachine will perform its initialization during which it executes an initial compound transition prompted by the creation, after which it enters a **wait point**. In case of StateMachine Behaviors, a wait point is represented by a stable state configuration. It remains thus until an Event stored in its event pool is dispatched. This Event is evaluated and, if it matches a valid Trigger of the StateMachine and there is at least one enabled Transition that can be triggered by that Event occurrence, a single StateMachine **step** is executed. A step involves executing a compound transition and terminating on a stable state configuration (i.e., the next wait point). This cycle then repeats until either the StateMachine completes its behavior or until it is asynchronously terminated by some external agent.
  - and trace classification in interactions:
    - Clause 13, Common Behaviors, describes the general semantics of the execution of Behaviors. Interactions are kinds of Behaviors that model emergent behaviors, as defined in sub clause 13.1. As discussed in sub clause 13.2.3, the execution of a Behavior results in an execution trace. Such a trace is a sequence of event occurrences, which, in this clause, will be denoted \(e_1, e_2, \ldots, e_n\). Each event occurrence may also include information about the values of all relevant objects at the point of time of its occurrence.
      The semantics of an Interaction are expressed in terms of a pair \([P, I]\), where \(P\) is the set of valid traces and \(I\) is the set of invalid traces. \(P\) need not be the whole universe of traces. Two Interactions are equivalent if their pairs of trace-sets are equal. The semantics of each construct of an Interaction (such as the various kinds of CombinedFragments) are

- **Model in standard libraries.**
Benefits: Classification Semantics

- **Standard execution models for UML** (fUML, etc)
  - Procedures that create a behavior occurrence
    - Conforming to a UML model.
  - Don’t tell whether
    - An existing behavior occurrence conforms.
    - Tools are producing correct occurrences

- **Classification does the opposite**
  - Tells whether an existing behavior occurrence conforms to a model.
  - Doesn’t say how to create an occurrence.
    - Execution engines and reasoners do this.
  - Enables semantic conformance testing.
Overview

- RoadMap
- Motivation
  - Behavior, review
  - 4D, requirements
- 4D Solution
  Objects and behaviors
  1. in space-time
  2. within time intervals, space regions
- Summary
“3D/4D” Modeling Defined

- “3D” = objects at an instant in time.
  - But no particular one (the “current” one).
- “4D” = objects as single entities extending over time.
  - Can be “sliced” up into intervals, including instants.

Not actually about physical space.
  - Includes behaviors treated like objects.
  - Better name: Snap / Span (Smith).
“3D” (Snap)

- Object “repeated” over time.
“4D” (Span)

- One object “through” time.
4D (space-time)

- An object “moves through” time in the same way as space.
  - Horizontal = car is stationary

Adapted from Matthew West, Developing High Quality Data Models, Morgan Kaufmann, 2011.
“3D / 4D” / 4D Compared

- “4D” is more general than “3D”.
  - “4D” covers all instants in an object lifetime, including the “current” one.
- “3D” can’t easily model:
  - Characteristics of objects over some of their lifetimes (including particular instants).
  - Can’t aggregate over time (eg, number of owners a car has had).
- 4D is more general than “4D”
  - 4D includes space.
UML: Not much “4D”

- **Instance specifications**
  - “Snapshots” can’t be related to same instance.
  - Don’t represent any particular instance, or any.

- **State invariants**
  - State machines and interactions.
    - On SM state or interaction fragment
    - Left to constraint language.
      - Holds for duration of state or interaction fragment
    - Can’t aggregate over time.
  - SM and interaction models aren’t related.
    - Just happen to have the same name.
4D Requirements

1. Treat objects and behaviors as existing in time and space together.
   – Behaviors in space of objects involved in them.

2. Specify objects and behaviors over
   – (Sub) intervals of time and regions of space.
Overview

- RoadMap
- Motivation
  - Behavior, review
  - 4D, requirements
- 4D Solution
  Objects and behaviors
  1. in space-time
  2. within time intervals, space regions
- Summary
Without and Within

- Completely separate or overlapping
  - Different dimensions (time, space) and terms.
  - Same thing (without, within).
Without and Within (M1)

- Without = happensBefore or outsideOf or both
- Within = Both happensDuring and insideOf
Objects and behaviors might have different names for 4D relations
- But the meaning is the same.
Overview

- RoadMap
- Motivation
  - Behavior, review
  - 4D, requirements
- 4D Solution
  Objects and behaviors
  1. in space-time
  2. within time intervals, space regions
- Summary
Time Slices

- All space in 4D region during time intervals.

Adapted from Matthew West, Developing High Quality Data Models, Morgan Kaufmann, 2011.
- All time in 4D region during (relative) space intervals.
Composition (Integral)

- Composite aggregation ("black diamond")

Wholes and parts are different kinds of things (cars and wheels)
Can take parts out and put them back
Composition (Portional)

- Same applies to slicing 4D things.
- New notation: 🌐

Portions are completely within their wholes.

Time and space slices are portions.
Time Slices (M1)

Standard model library

Model (M1)

System model

Occurrence

happens Before

timeSlice

Car

structure

parked :

happensBefore

operated :

Time slices of cars are also cars (could be subtypes)

Instances (M0)

John’s Vehicle
May 15, 2007 – Sep 12, 2016

John’s Vehicle (Parked)
Jan 5-6, 2015, 7pm-8amET
: Car

John’s Vehicle (Operated)
Jan 6, 2015, 8-8:30amET
: Car
Time Slices (M2)

Metamodel (M2)

Model (M1)

Standard model library

System model

Class

Occurrence

Property

OState Property

Occurrence Property

owned attribute

happensBefore

structure

parked :

happensBefore

operated :

«ostate»

parked

*«ostate»

operated

*
- Car should only be in road when it’s clear.
4D, Multi-object (M1)

- **Car Operating Environment**
  - **structure**
  - **path**
    - **Vehicle**: «ostate» moving
      - path Ahead
    - **Path**: «ostate» clear
      - happens During

- **Model (M1)**
  - **Car**: «ostate» moving, «ostate» parked
  - **Road**: «ostate» clear
  - **Parked Car**: without

- **Notes**:
  - Portion of road where car is going.
  - Cars can move only when path ahead is clear.
  - Cars can’t park in the road.
Snapshots

- Times slices with zero duration.
Snapshots (M1)

- Snapshots happen before themselves.
  - They “meet” at beginning and end.
Start/Endshots

- Snapshots at beginning and end of 4D occurrences (including slices).
Start/Endshots happen before/after all snapshots.
Objects might have different names for start and end
- But the meaning is the same.

Behaviors might specialize for end status
Two links between John’s car
– One before engine is removed and one after.

Links between John’s car and its engine

- Car-engine link #1
- Car-engine link #2
Links (M1 & M2)

Metamodel (M2)

Standard Model Library

Model (M1)

System Model

Instances (M0)

M1 property at tail of arrow is value of M2 property at head of the arrow.
*Not instance links*
Link Occurrences (M1)

Standard Model Library

Model (M1)

System model

Instances (M0)

John's Vehicle
5/1/5/2007-9/12/2016:

Engine
5/1/5/2007-9/12/2016:

Powers
6/28/5/2010-9/12/2016:

5/1/5/2007-6/23/2010:
Transfers (M1)

Model (M1)
- Standard Model Library
- System Model

Instances (M0)
- Product Transfer 3/15/09 10-12pmET:
  - Stove234:
    - source: Store654:
    - target: John’sHouse:
Transfers as Links (M1)

Standard Model Library

Model (M1)

System Model

Instances (M0)

Product Transfer

3/15/09 10-12pm ET:

Source: Store654:
Target: John’s House:

Product Transfer

Stove234:

Transfer

Product

BinaryLink

Occurrence

Item

Source

Target

Item

[1..*]

(overrides)

[1..*]

source [1]

target [1]

target [1]

source [1]

Item

Item
Transfers & I/O (M2)

Metamodel (M2)

Model (M1)

Instances (M0)

**TakePicture** Activity

- **step1**: Focus
  - **out xrsI**: Exposure

- **step2**: Shoot
  - **in xfs**: Exposure

**TakePicture 3/15/09 10-12pmET**:

- **Focus Occ 1**: -> **Shoot Occ 1**
  - **: ExposureTransfer**
  - **: HappensBefore**

**Class**

**Item Flow**

- **sourceOutputProperty**
  - **[1..*] {ordered, non-unique}**

- **targetInputProperty**
  - **[1..*] {ordered, non-unique}**

**Property**

**Metadata**

- **itemType**
  - **[1..*] {ordered, non-unique}**

**Connector**

**Flow**

**M1 property at tail of arrow is value of M2 property at head of the arrow.**

*Not instance links*
Transfer & I/O Semantics

1. Transferred item = value of
   – source output property when transfer starts.
   – transfer item property during transfer.
   – target input property when transfer ends.

2. Transfer might:
   – remove item from source output property when it starts (isMove).
   – start when item is added to source output property (isPush)
   – have zero duration (isInstant)
Transfer & I/O Semantics (#1)

Connector typed by source output association
(on specialization or item flow)

Connector typed by target input association
(on specialization or item flow)

Standard Model Library (M1)

**item** = value of:
- **source output property when transfer starts**
- **item property during transfer**
- **target input property when transfer ends**
Transfer & I/O Semantics (#2)

Transfer

- `isMove` : Boolean = true {readonly}
- `isPush` : Boolean = true {readonly}
- `isInstant` : Boolean = true {readonly}

```
source 1
       «isSufficient» : BinaryLink
                      item 1..*
                      «isSufficient» : BinaryLink
                     target 1
```

- `sourceOutputLink [1..*]`
- `targetInputLink [1..*]`

- `startShot`
- `endShot`

- `{ isPush }`
- `{ isMove }`
- `{ isInstant }`

- Remove item from source output when transfer starts
- Start transfer when item is added to source output
- Transfer ends when item is added to target input
- Transfer has zero duration

Standard Model

Library (M1)
4D TBD

- More about space
- Formalize slicing
- Quantification
  - Units
Overview

- RoadMap
- Motivation
  - Behavior, review
  - 4D, requirements
- 4D Solution
  Objects and behaviors
  1. in space-time
  2. within time intervals, space regions
- Summary
Summary

- Unify objects and behaviors as occurrences
  - occupying regions of space-time.
- Unify spatial and temporal relations as
  - relations between regions of space-time.
- Introduce portion composition
  - Specialize for slices of time and space
  - Specialize time slices for snapshots.
  - Specialized snapshots for start and end shots.
- Treat links as occurrences
  - Apply to model item flow semantics.
Past ADTF Intro Slides

- **Intro to Behavior as Composite Structure**


- **Object-orientation**: [http://doc.omg.org/ad/18-09-07](http://doc.omg.org/ad/18-09-07)

- **State Machines, parts 1&2:**
  - [http://doc.omg.org/ad/18-12-09](http://doc.omg.org/ad/18-12-09)
  - [http://doc.omg.org/ad/19-03-02](http://doc.omg.org/ad/19-03-02)

- **Activities, part 2**: [http://doc.omg.org/ad/19-06-02](http://doc.omg.org/ad/19-06-02)
More Information

- Earlier slides (more onto)

- Papers:
  - Ontological Behavior Modeling:
    http://dx.doi.org/10.5381/jot.2011.10.1.a3
  - Ontological Product Modeling:
    https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=822748
  - 4D Requirements Modeling:
    https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=919164

- Application to BPMN:
  http://conradbock.org/#BPDM

- SysML2: Contact Bjorn Cole bjorn.f.cole@lmco.com