Constraint-enabled Process Modeling

Conrad Bock

U.S. National Institute of Standards and Technology

November 20, 2007

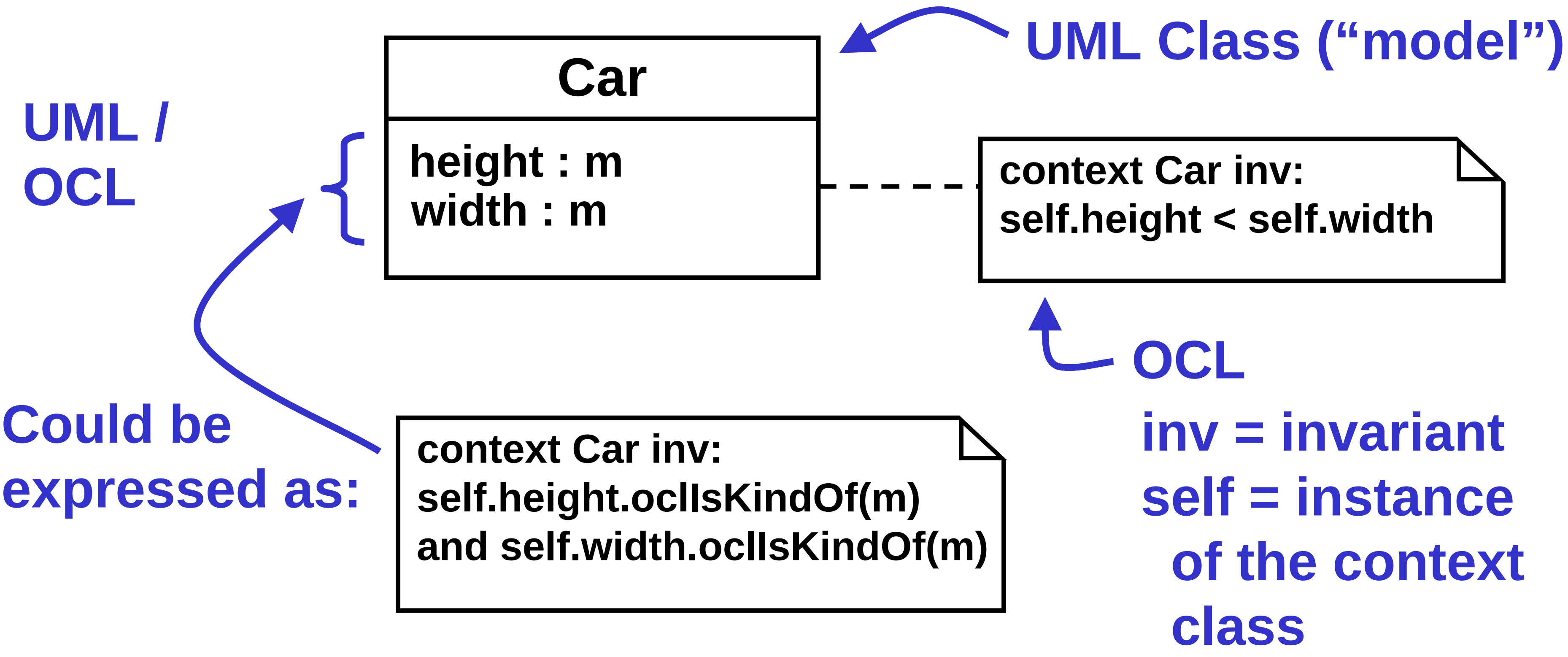
Overview

- Models and constraints: Example of structure models
- Extend to process models: execution
- Process modeling that includes execution
- Execution constraints
- Process Specification Language (PSL)
- Relating process / execution modeling to PSL

Structure Models and Constraints

- Structural modeling languages have associated constraint languages:
  - UML includes OCL
  - OWL used with SWRL
  - RDF used with SPARQL
  - EXPRESS includes EXPRESS rules
- Simple, commonly used statements in modeling language, more detail in constraint language.

Structure Models are Shorthands



UML Class (“model”)

Car

height : m
width : m

UML / OCL

context Car inv:
self.height < self.width

OCL

inv = invariant

self = instance of the context class

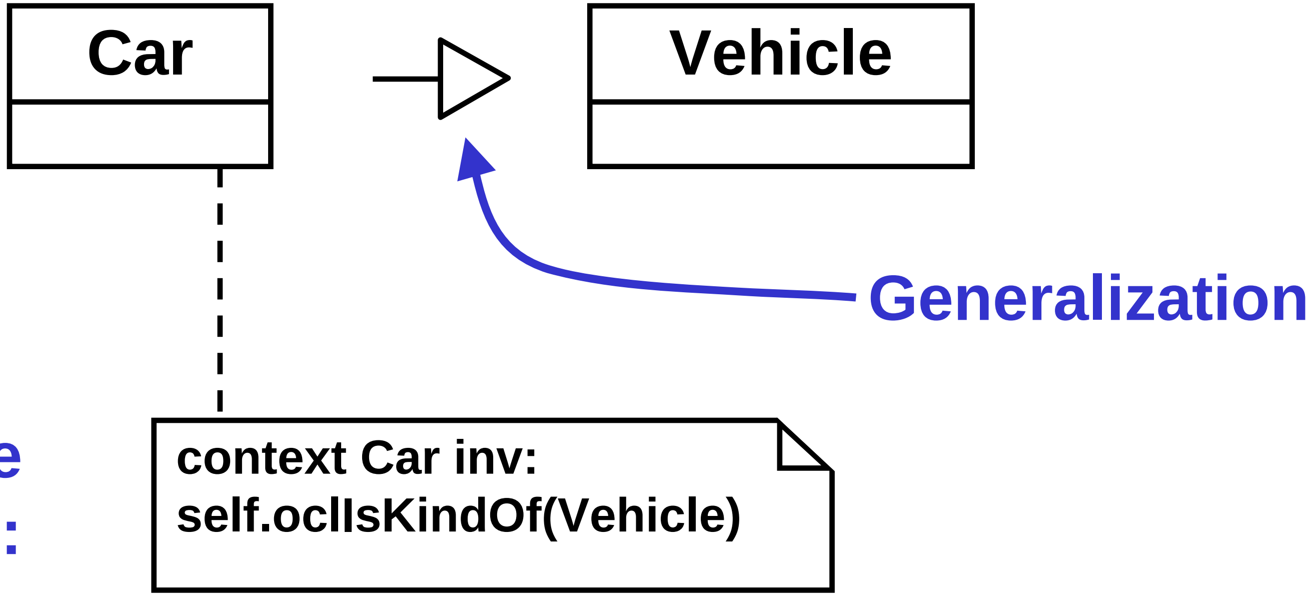
Could be expressed as:

context Car inv:
self.height.oclIsKindOf(m)
and self.width.oclIsKindOf(m)

- Could write entire model in OCL, just a matter of ergonomics.
- Enables structural models to have constraints.

Structure Models are Shorthands

UML



Car

Vehicle

Generalization

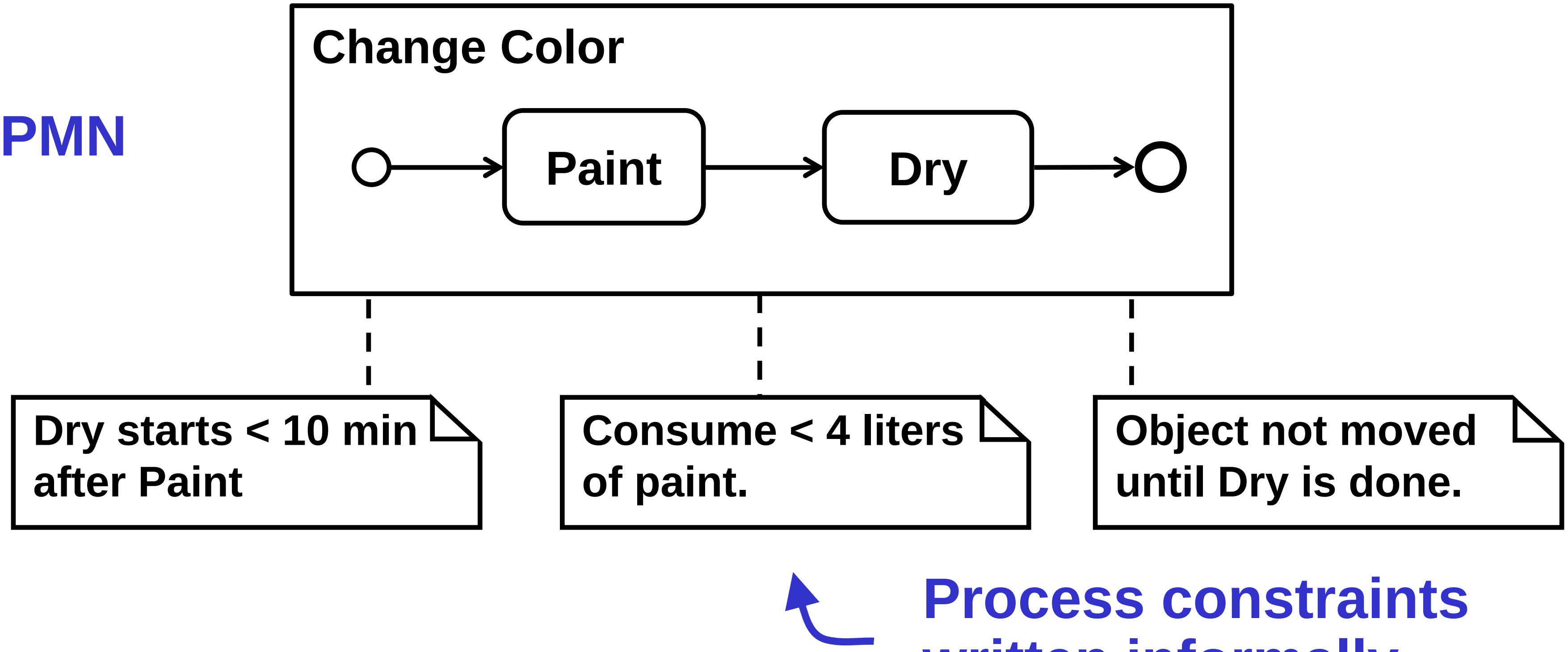
Could be expressed as:

```
context Car inv:
self.oclIsKindOf(Vehicle)
```

- Generalization = all instances of subtype are instances of supertype (“cars are vehicles”).
- Constraints on (instances of) supertype apply to (instances of) subtype.

Constraints for Process Models?

BPMN

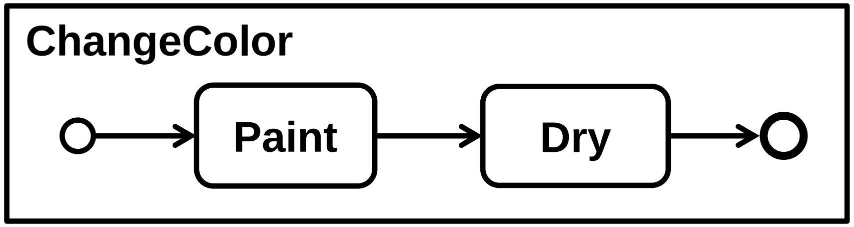

Change Color
Paint
Dry
Dry starts < 10 min after Paint
Consume < 4 liters of paint.
Object not moved until Dry is done.

Process constraints written informally

- Specify additional details on process models.
- Could be about timing, resource consumption, service level agreements, etc.

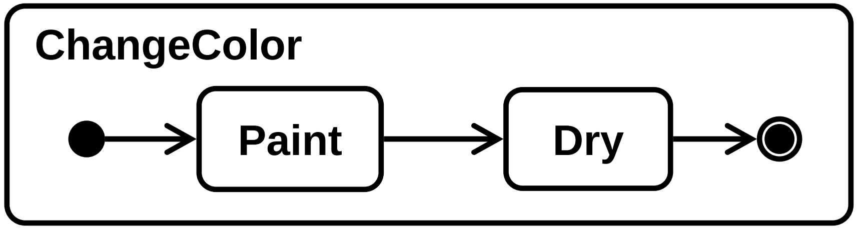
Are Process Models Shorthands?

- BPMN:



(or XPDL)

- UML 2:



(or XMI, repository)

- BPEL:

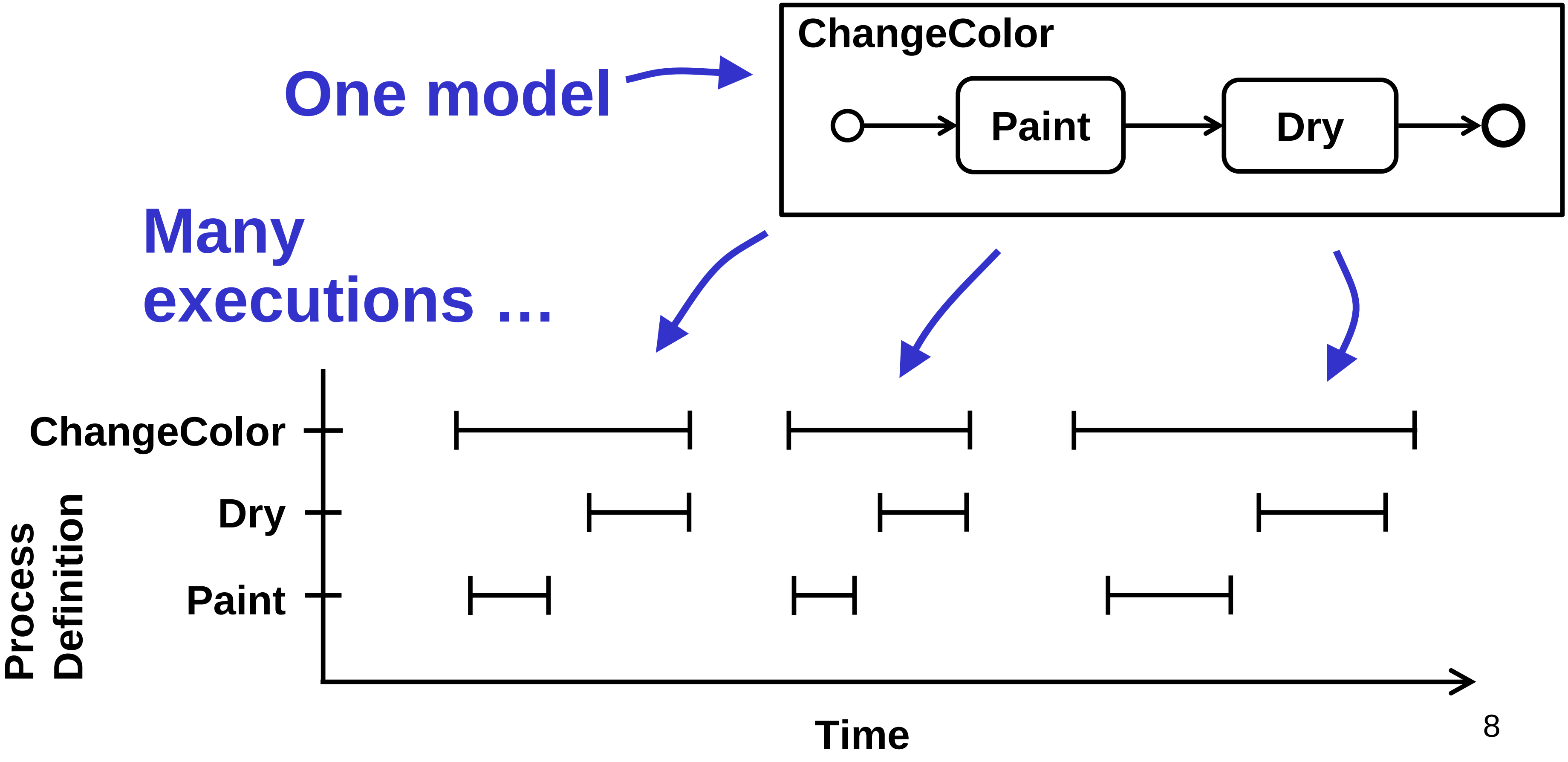
```
<process name=“ChangeColor”>
  <sequence>
    <invoke operation=“Paint”></invoke>
    <invoke operation=“Dry”></invoke>
  </sequence>
</process>
```

- Java/C:

```
void ChangeColor
{ Paint();
  Dry();
}
```

What's Constrained?

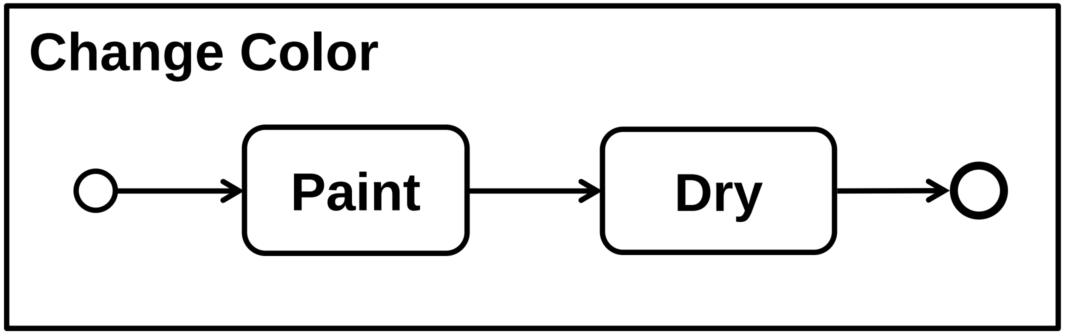
- "Instances" of processes.
- Executions



Execution Tools

- Workflow systems call executions “process instances”.
- Process management systems provide
  - monitoring, analysis
  - other services on executions.
- These systems treat executions as first-class entities, with their own
  - attributes (eg, elapsed time)
  - operations (eg, suspend)

Execution Constraints



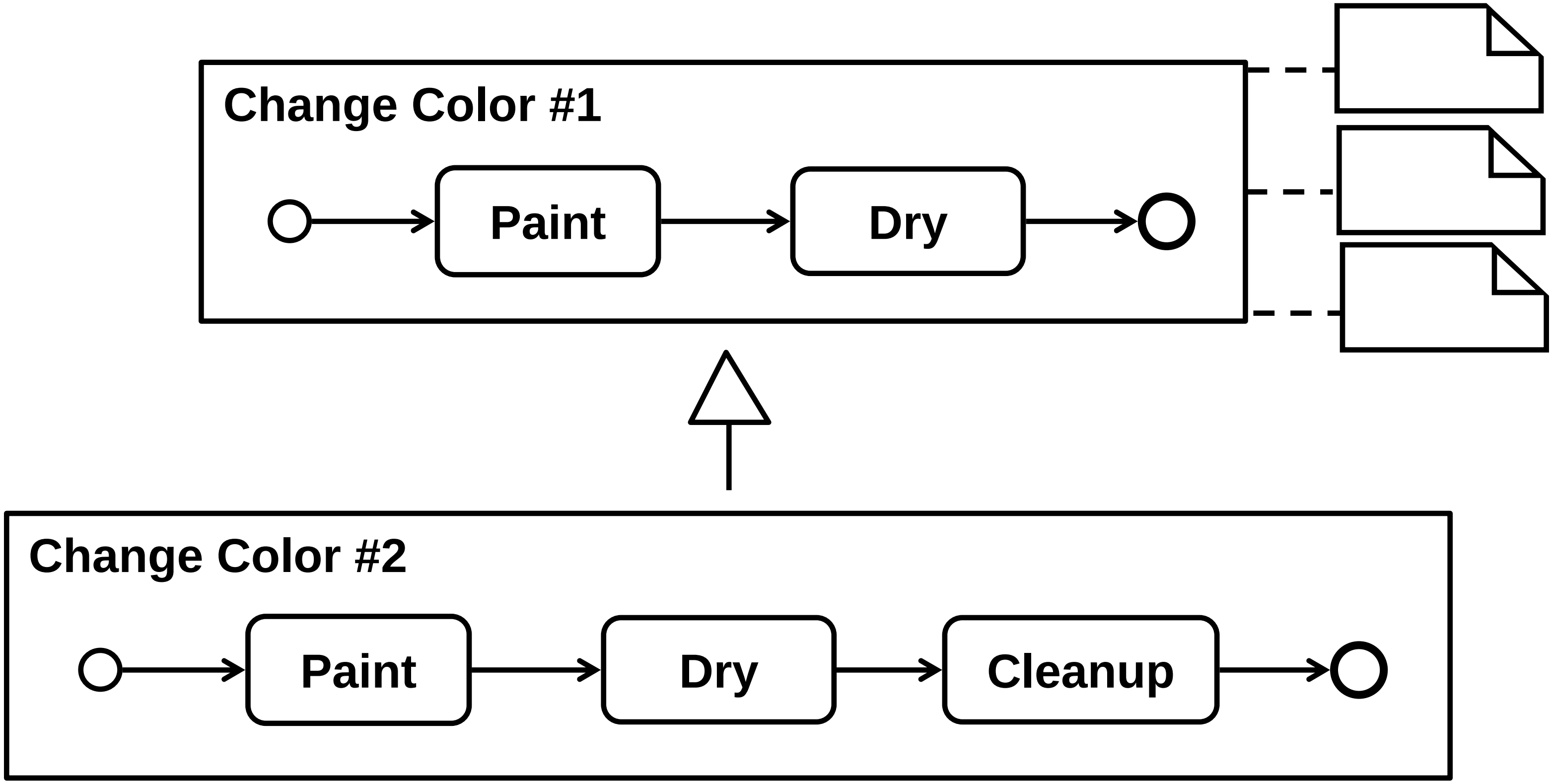
Could be expressed as:

For every execution of ChangeColor, executions of Paint and Dry occur, with the Paint execution happening before the Dry.

Process constraint written informally

- Could write entire model as execution constraints, just a matter of ergonomics.
- Would enable process models to have constraints.

Process Generalization



- All executions of specialized process are executions of generalized process.
- Constraints on (executions of) #1 apply to (executions of) #2.

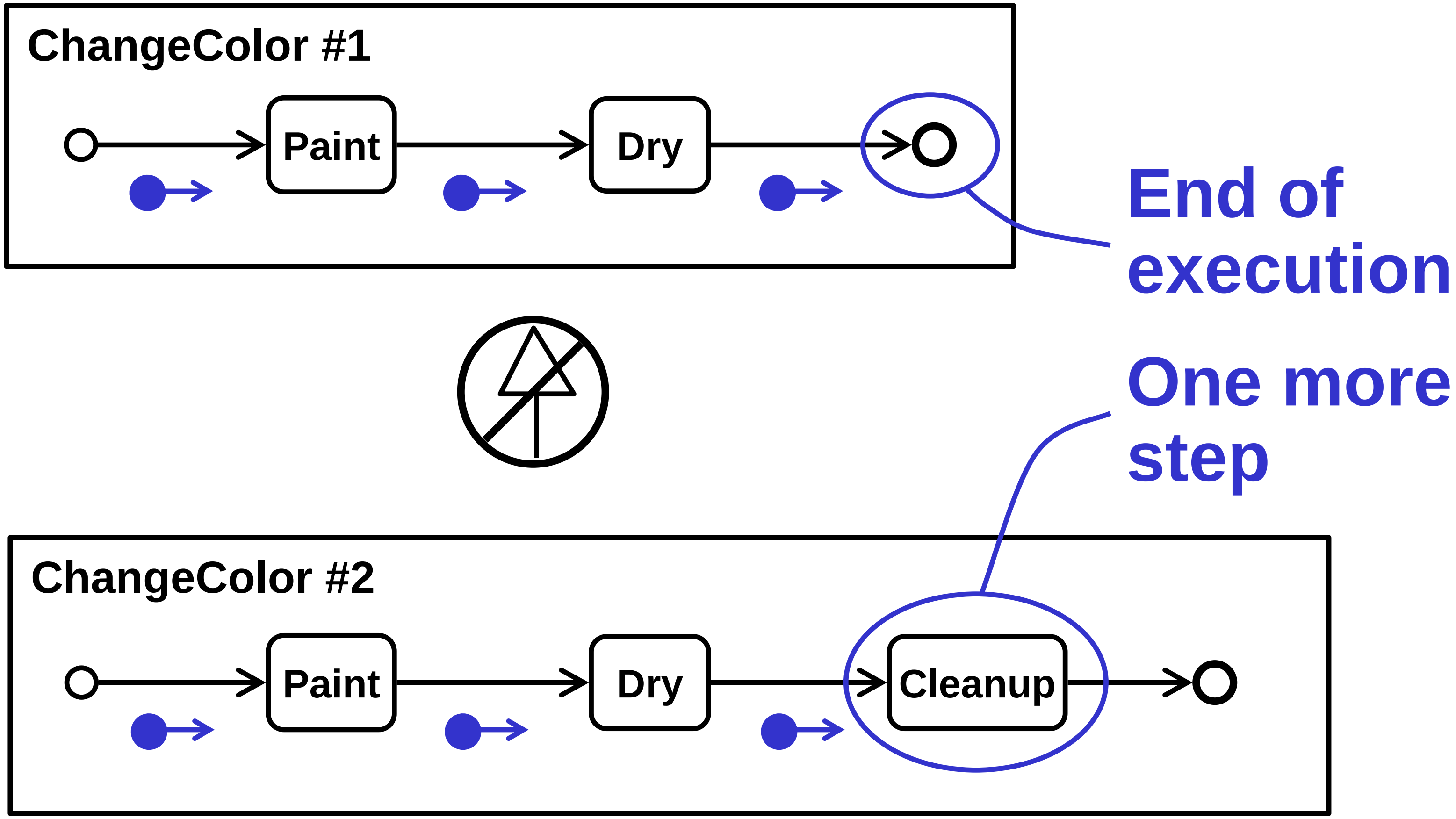
Semantics and Syntax

- **Semantics:**
  - **Need a semantics for process models that can be integrated with constraints.**
- **Syntax:**
  - **Need a constraint syntax that can be integrated with process models.**
- **This presentation is about semantics.**

Virtual Machine Semantics

- Common process models usually have semantics defined by an imagined or virtual machine that “executes” (interpreter).
- Produce executions given particular inputs or conditions.
- A kind of operational semantics.
- Does not directly support automated reasoning or abstraction.

Token “Movement”

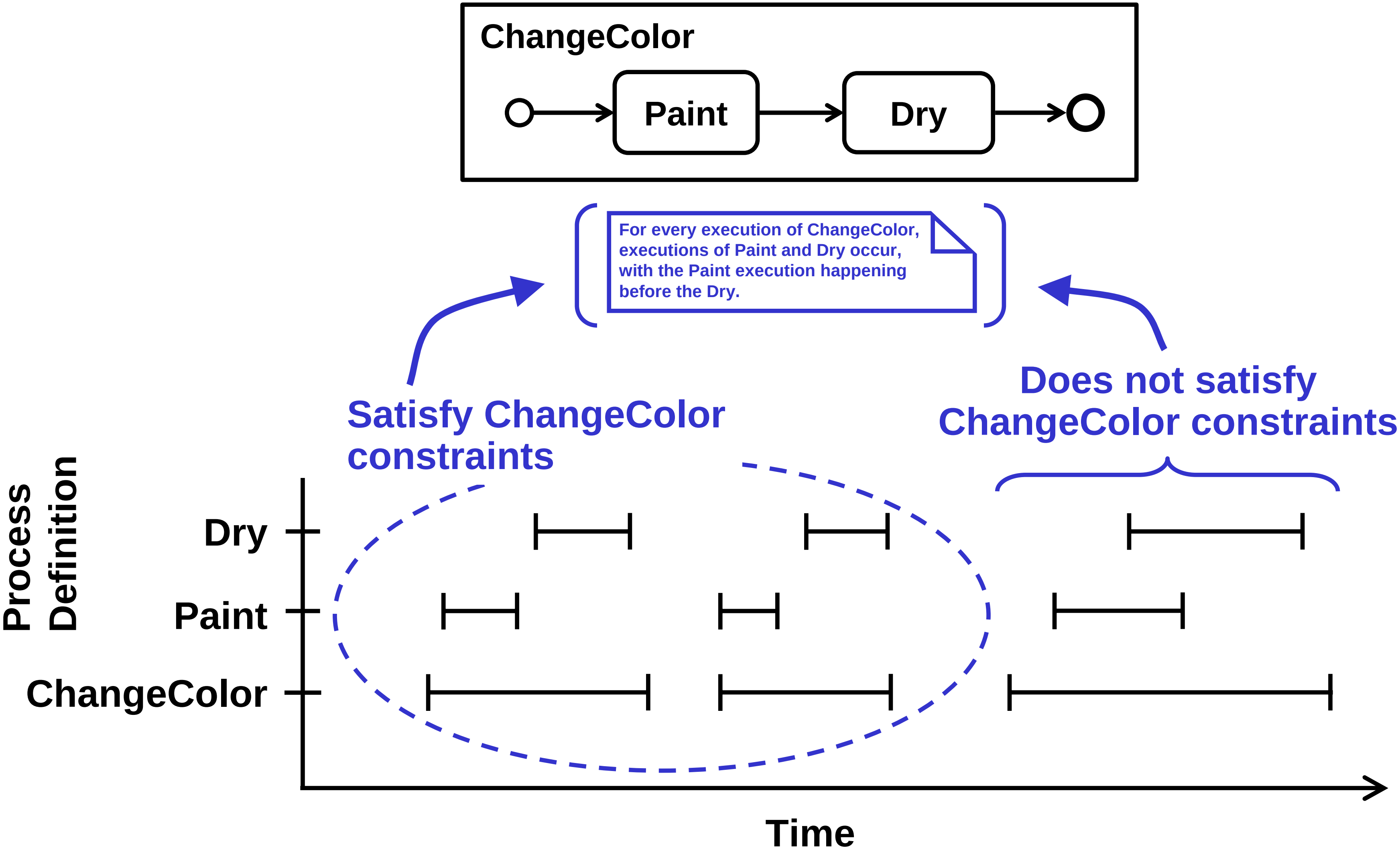


- **Executions satisfying #2 do not satisfy #1 under token semantics.**

Execution Constraint Semantics

- Tells which executions conform to (“satisfy”) process model.
- Does not directly produce executions, but infers the ones allowed by model.
- Supports automated reasoning and abstraction.
- A kind of axiomatic semantics.
- Might also be called “declarative”.

Execution Constraint Semantics



For every execution of ChangeColor, executions of Paint and Dry occur, with the Paint execution happening before the Dry.

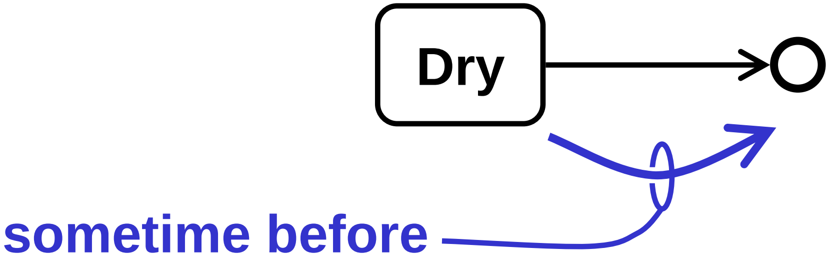
Satisfy ChangeColor constraints

Does not satisfy ChangeColor constraints

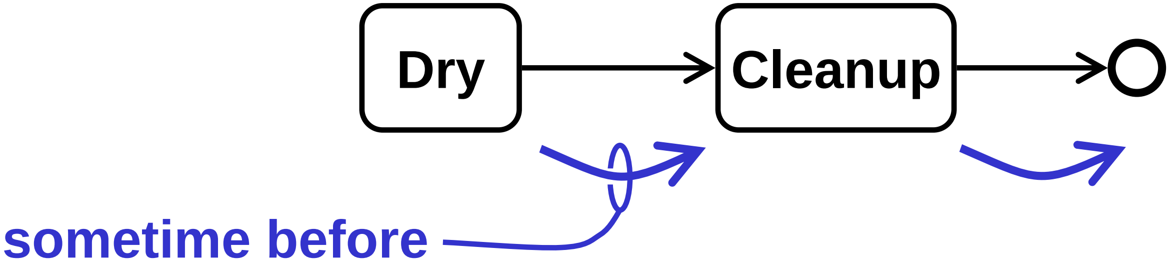
- Some executions satisfy the model, some do not.

Partial Ordering Constraints

- One step or message happens *sometime* before another, not necessarily immediately.
- Allows “insertions”:

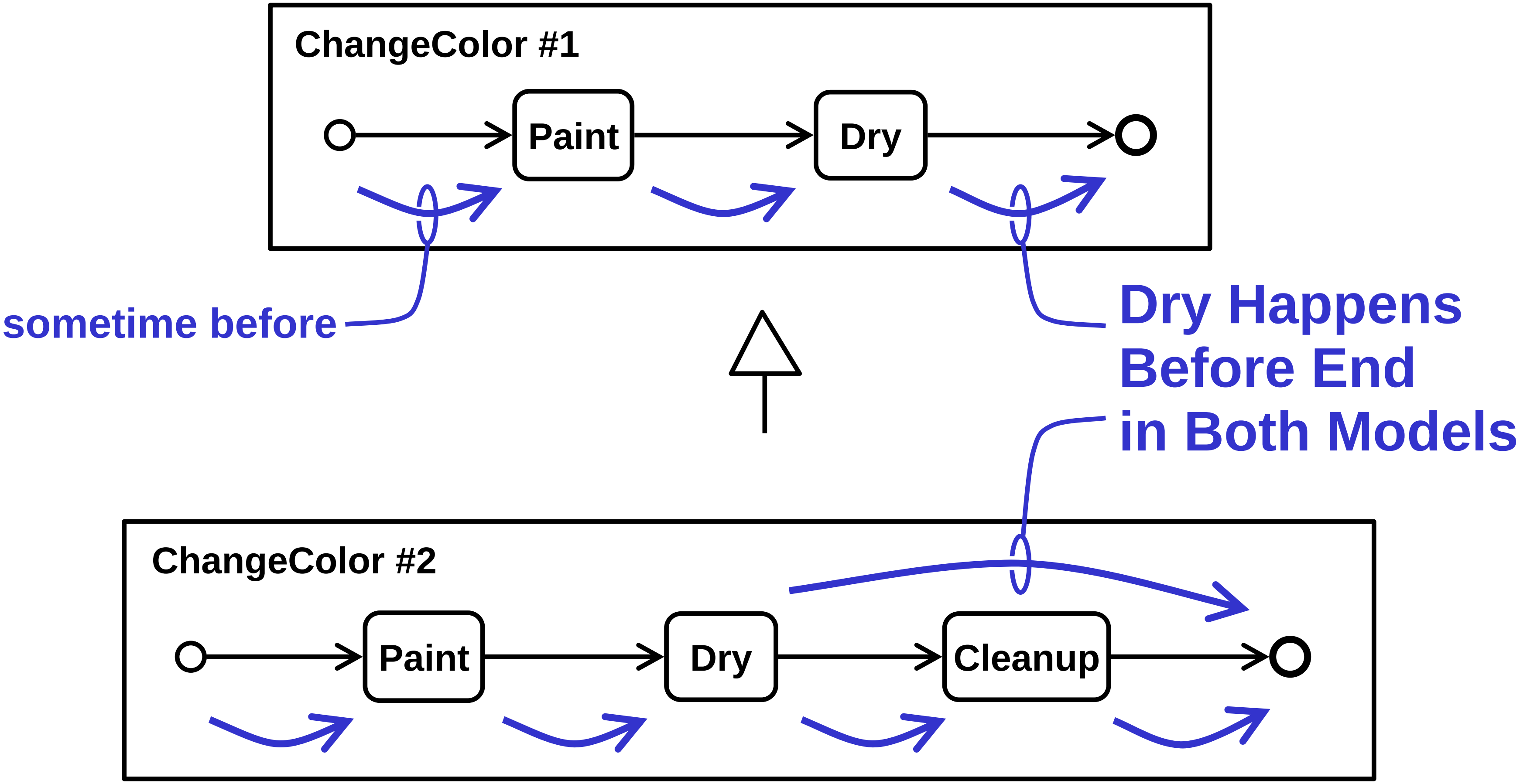


is consistent with



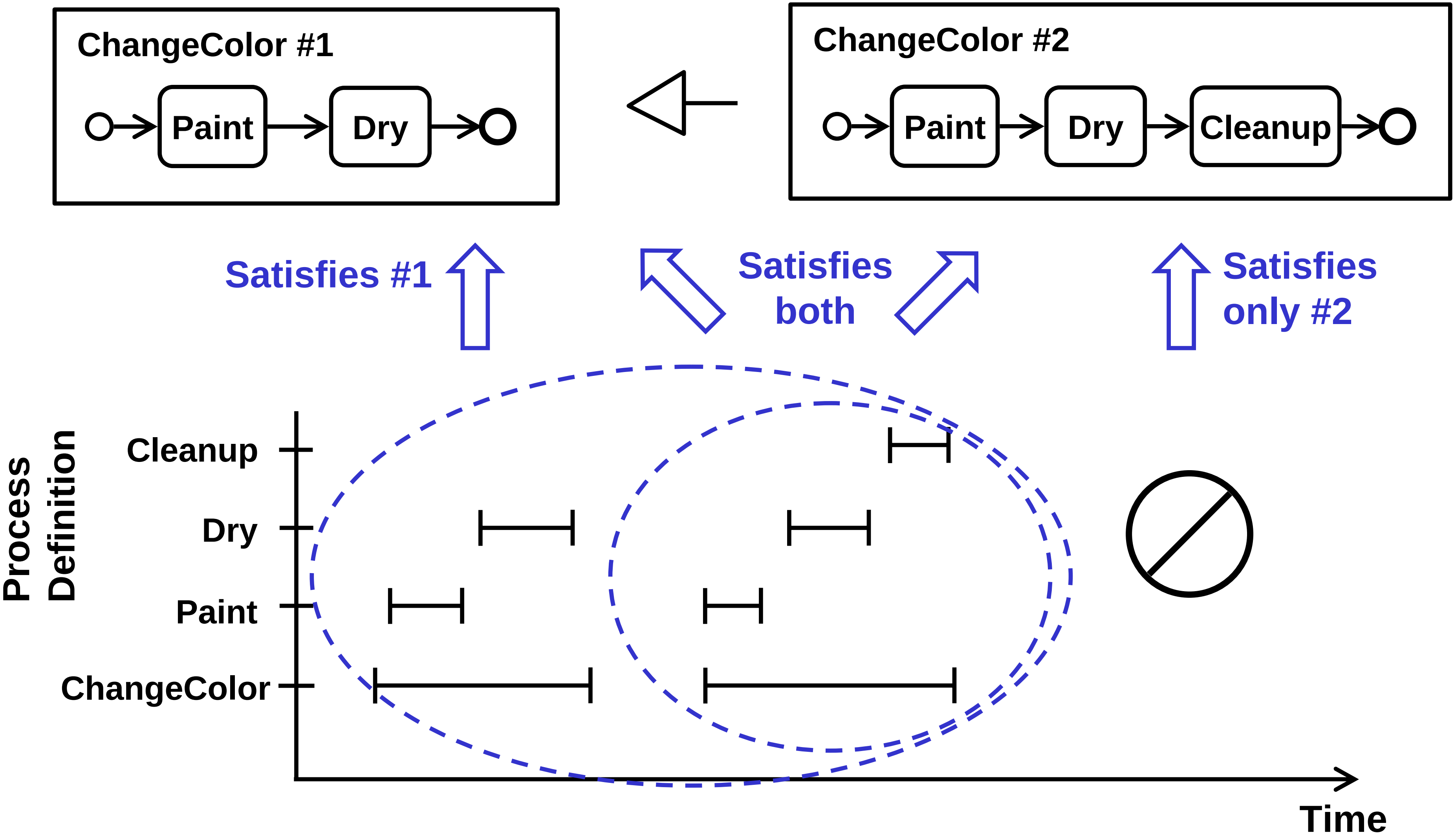
(partial ordering is transitive)

Specializing Partial Orders



- **Executions satisfying #2 also satisfy #1 under happens sometime before.**

Execution Subsets



ChangeColor #1

ChangeColor #2

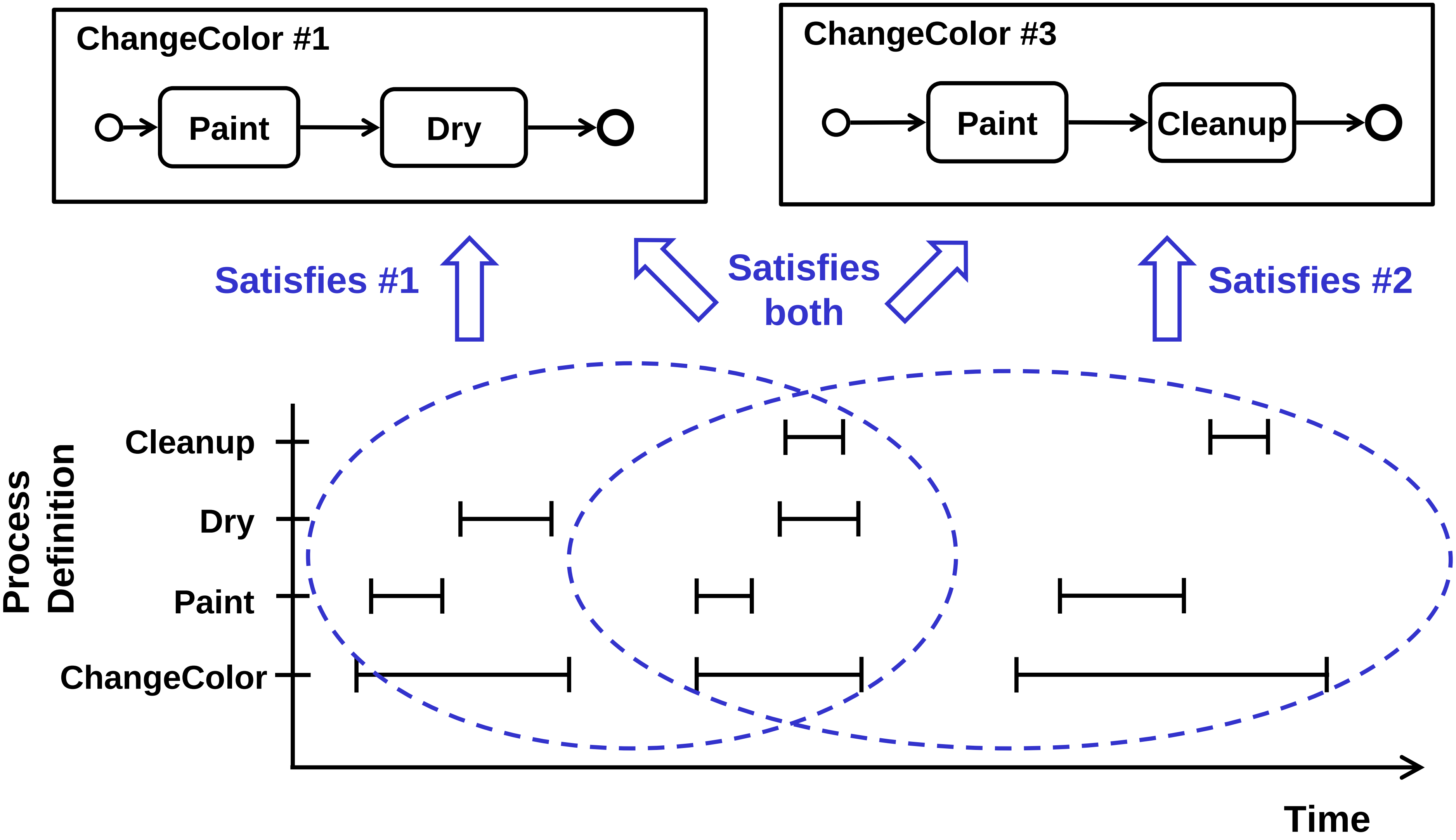
Satisfies #1

Satisfies both

Satisfies only #2

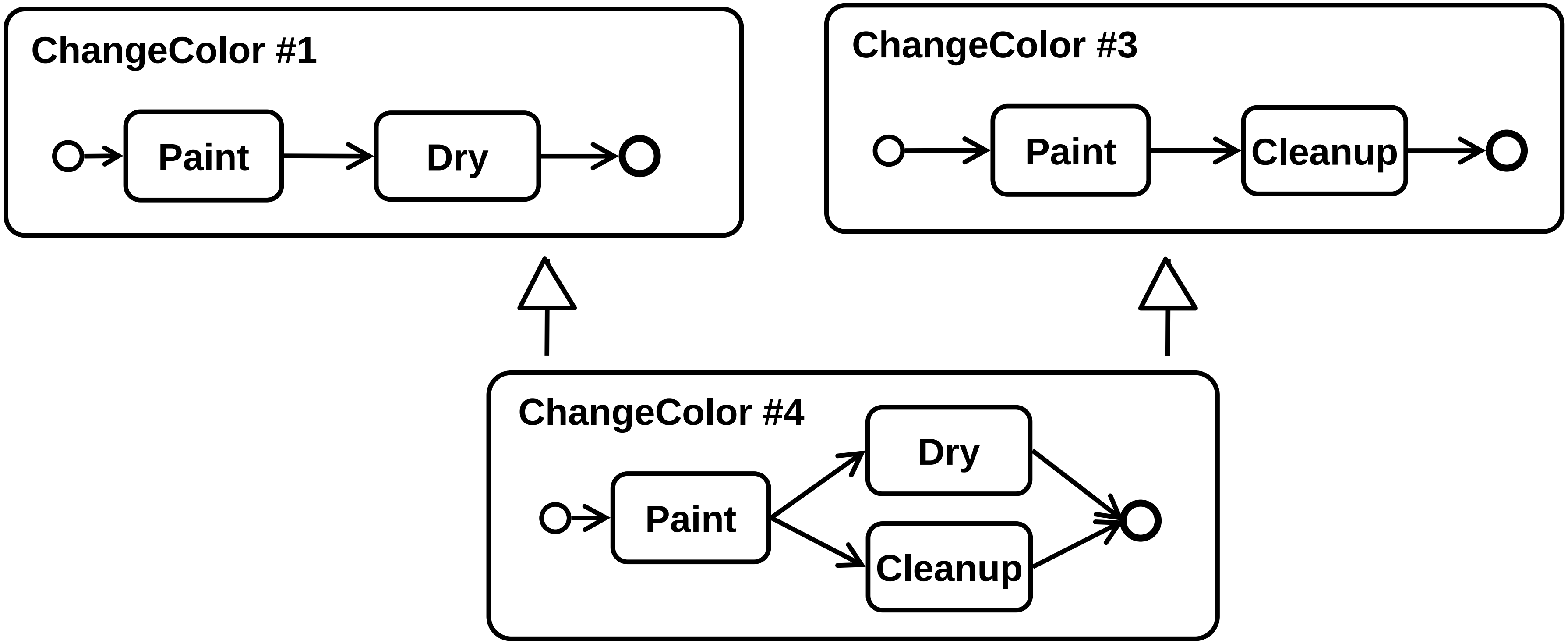
- **Executions satisfying #2 are a subset of executions satisfying #1.**

Overlapping Models



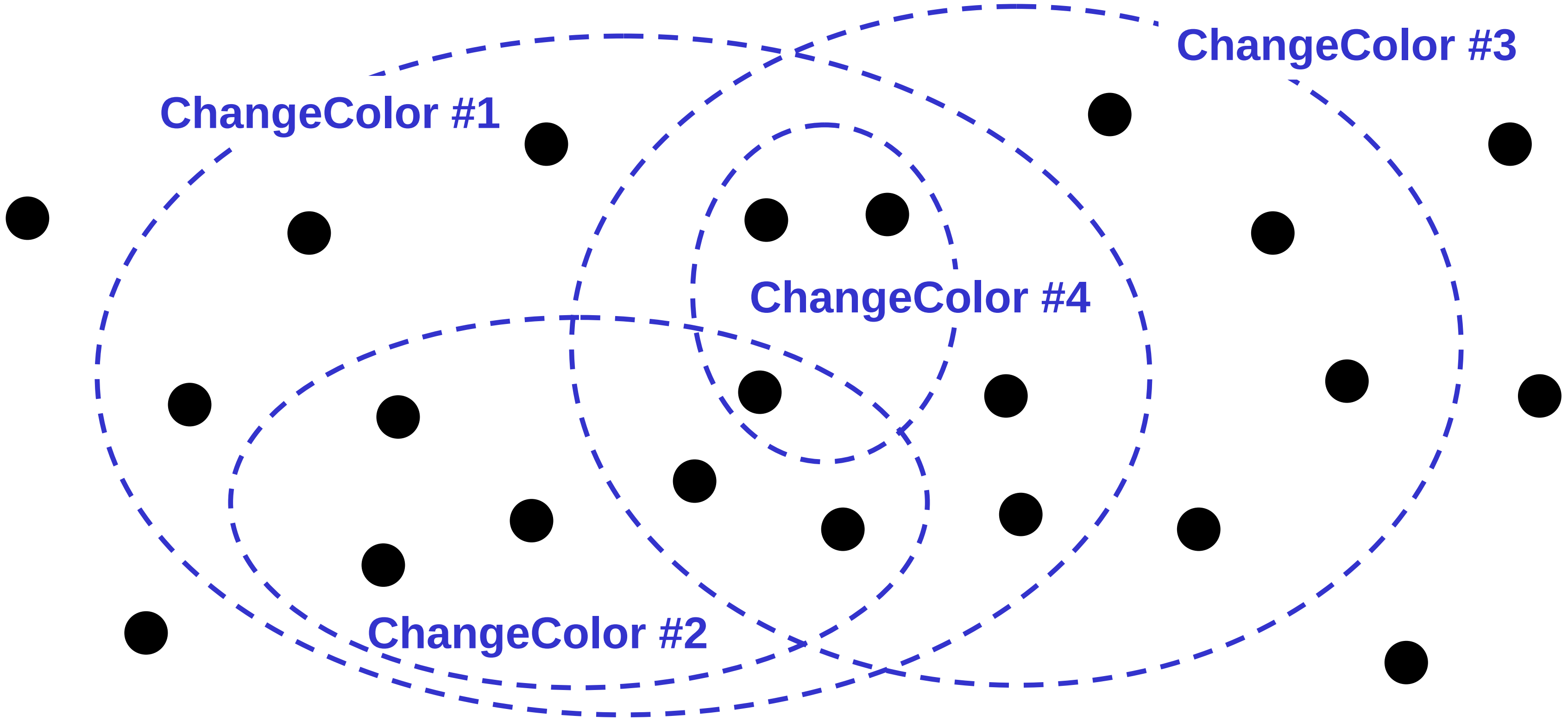
- Some executions in the “intersection”.

Overlapping Models



- **Executions satisfying #4 also satisfy #1 and #3, but some only satisfy #1 or only #3.**

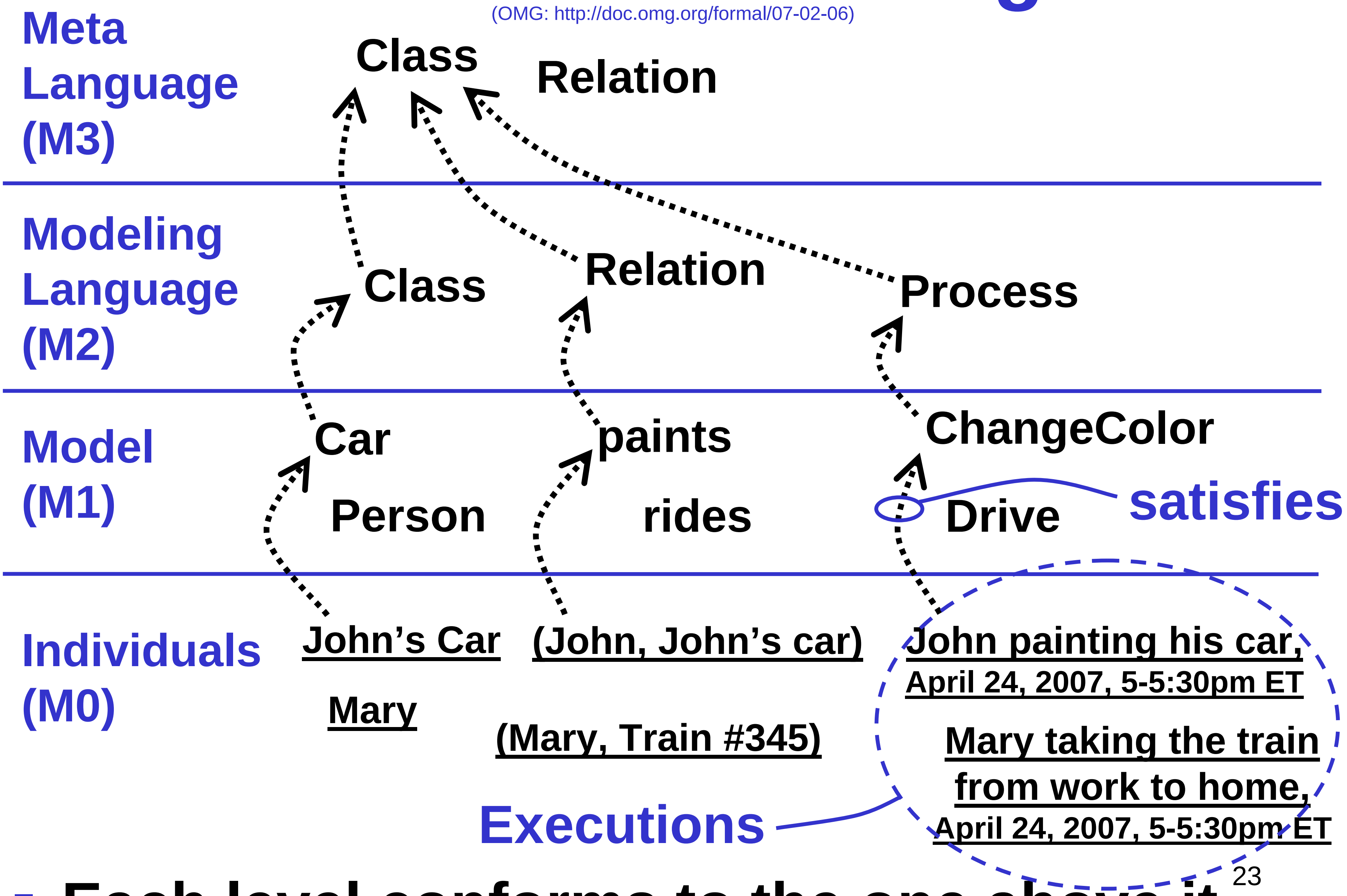
Venn Diagram of Examples



- ● = Execution

Levels of Modeling

(OMG: http://doc.omg.org/formal/07-02-06)



Meta Language (M3)

Class

Relation

Modeling Language (M2)

Class

Relation

Process

Model (M1)

Car

paints

ChangeColor

Person

rides

Drive

satisfies

Individuals (M0)

John's Car

(John, John's car)

John painting his car, April 24, 2007, 5-5:30pm ET

Mary

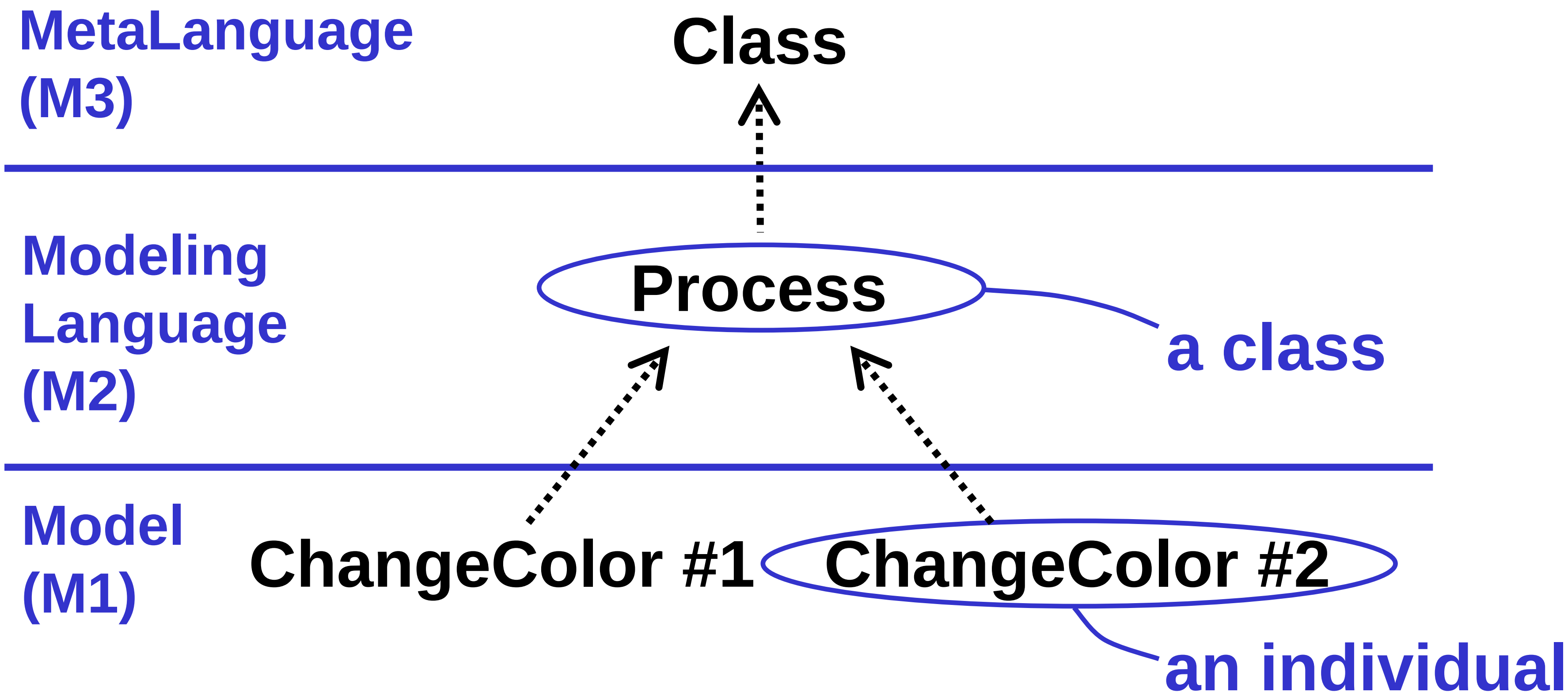
(Mary, Train #345)

Mary taking the train from work to home, April 24, 2007, 5-5:30pm ET

Executions

- **Each level conforms to the one above it.**

Modeling Without Execution



MetaLanguage (M3)

Class

Modeling Language (M2)

Process

a class

Model (M1)

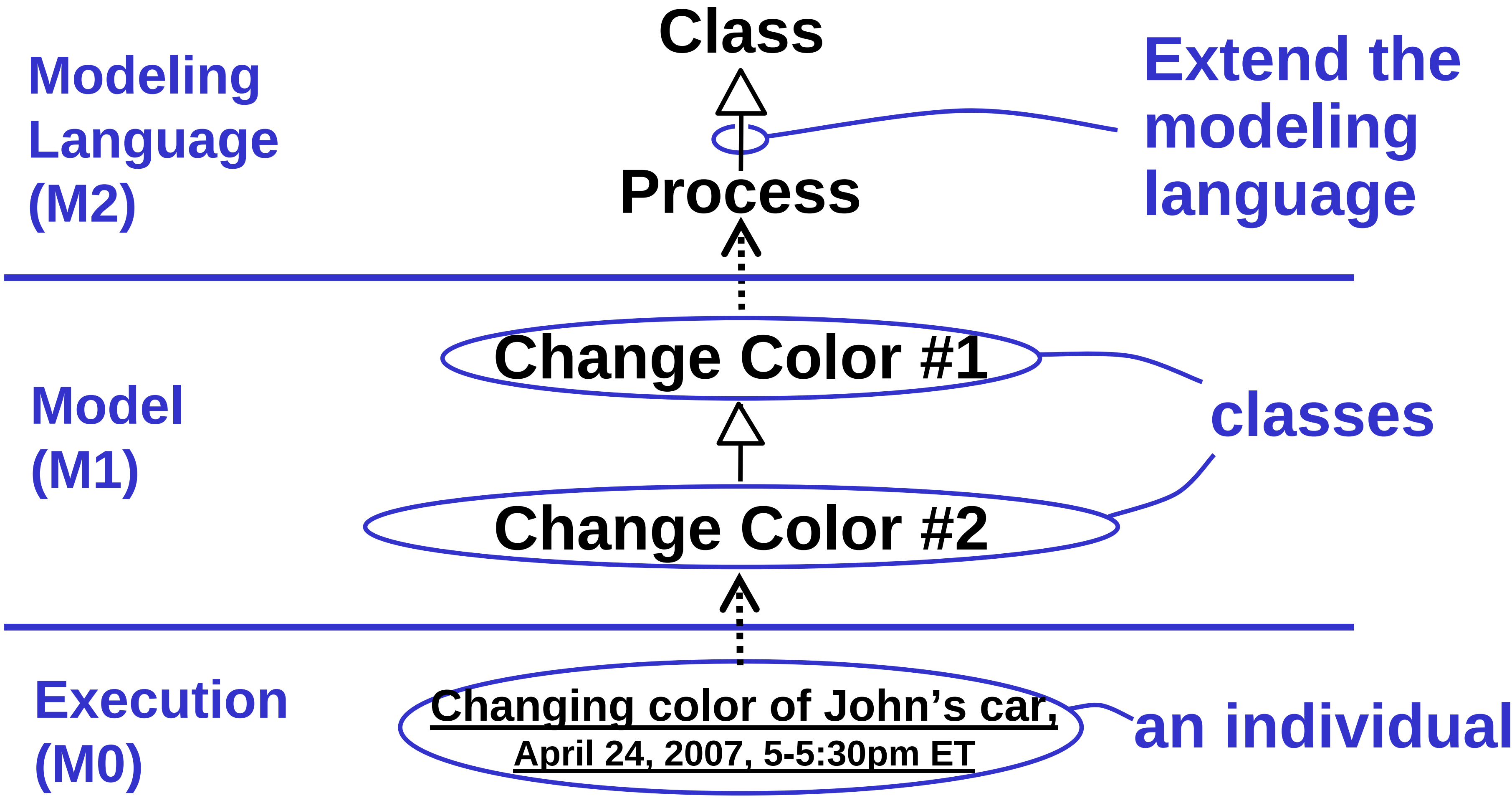
ChangeColor #1

ChangeColor #2

an individual

- Cannot instantiate and specialize user models (they are individuals, not classes).
- Unrelated to runtime execution (M0).

Modeling With Execution



Modeling Language (M2)

Class

Process

Extend the modeling language

Model (M1)

Change Color #1

Change Color #2

classes

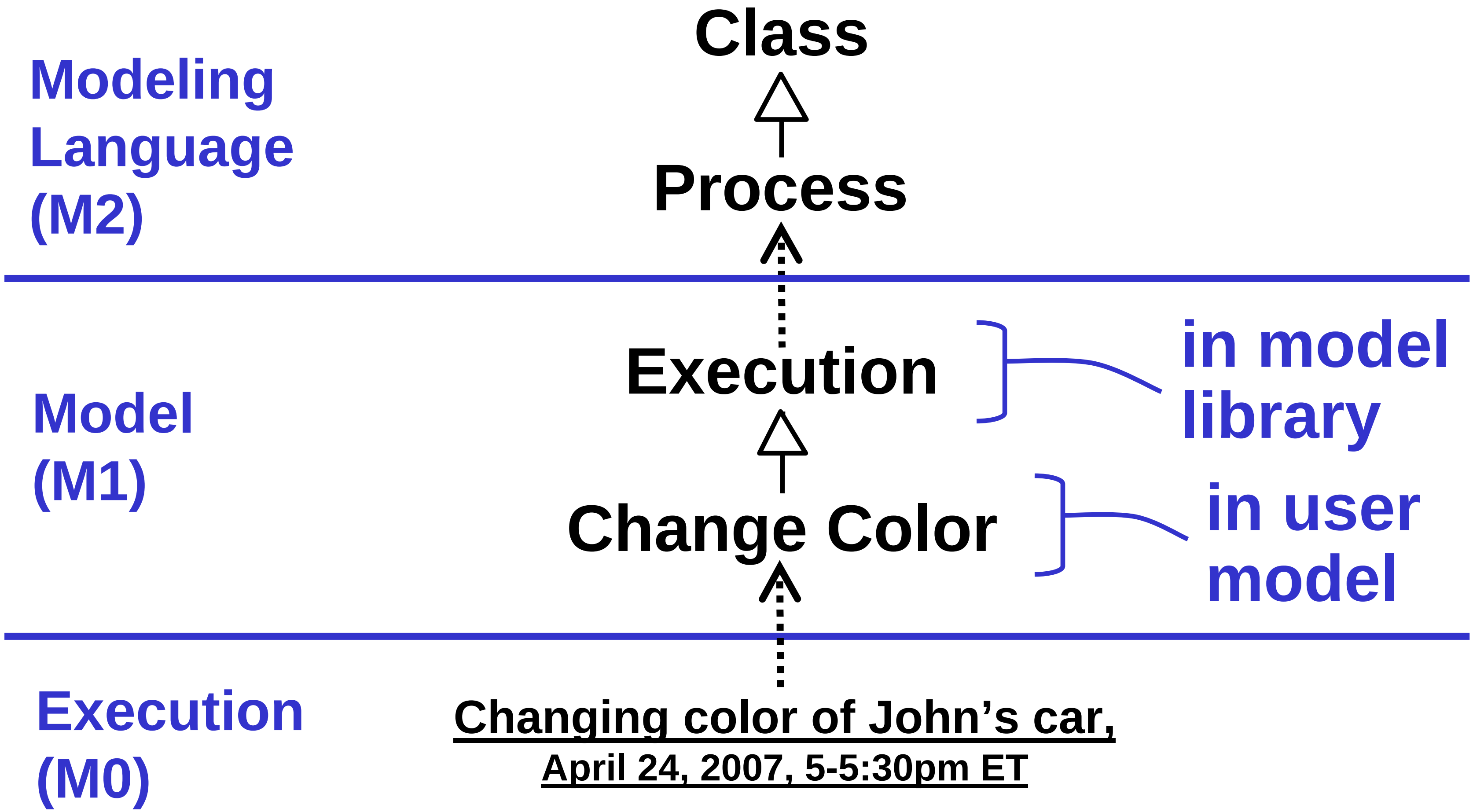
Execution (M0)

Changing color of John's car, April 24, 2007, 5-5:30pm ET

an individual

- **M1 process models are classes, can be specialized in M1 and instantiated at M0.**
- **M1 process constraints apply to M0 executions.**

Modeling With Execution



Class

Process

Modeling Language (M2)

Execution

in model library

Model (M1)

Change Color

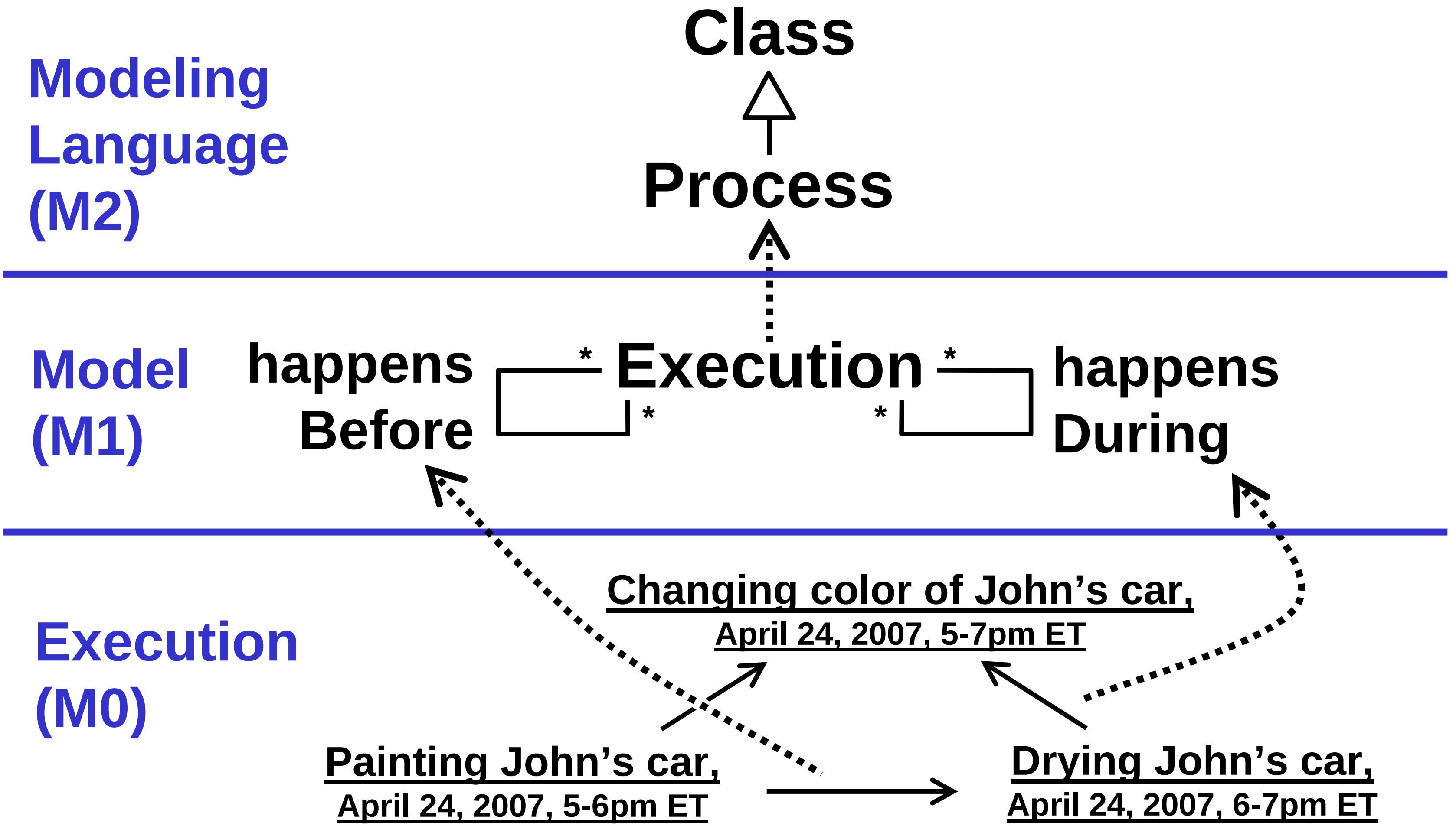
in user model

Execution (M0)

Changing color of John’s car, April 24, 2007, 5-5:30pm ET

- **Class of all executions. Superclass of all process models. Introduces attributes such as time elapased and resources used.**
- **(Universal Behavior in BPDM)**

Modeling With Execution



- The Execution class introduces temporal relations.
- Must limit happensBefore by happensDuring.

Execution Constraints

- **Can apply class-based constraint languages (like UML's OCL):**

```
context ChangeColor inv:
   self.paintStepDuringCC.happensBefore->
   includes(self.dryStepDuringCC)
```

- **Or first order (Common Logic Interchange Format):**

```
(forall (?CC ?P ?D)
   (if (and (ChangeColor ?CC)
            (paintStepDuringCC ?P ?CC)
            (dryStepDuringCC ?D ?CC))
       (happensBefore ?P ?D)))
```

•Variables are executions.
•Predicates are sets of executions or pairs of executions.

- **Common enough to be promoted to modeling languages (BPMD Succession, edges in typical graphical flow languages)**

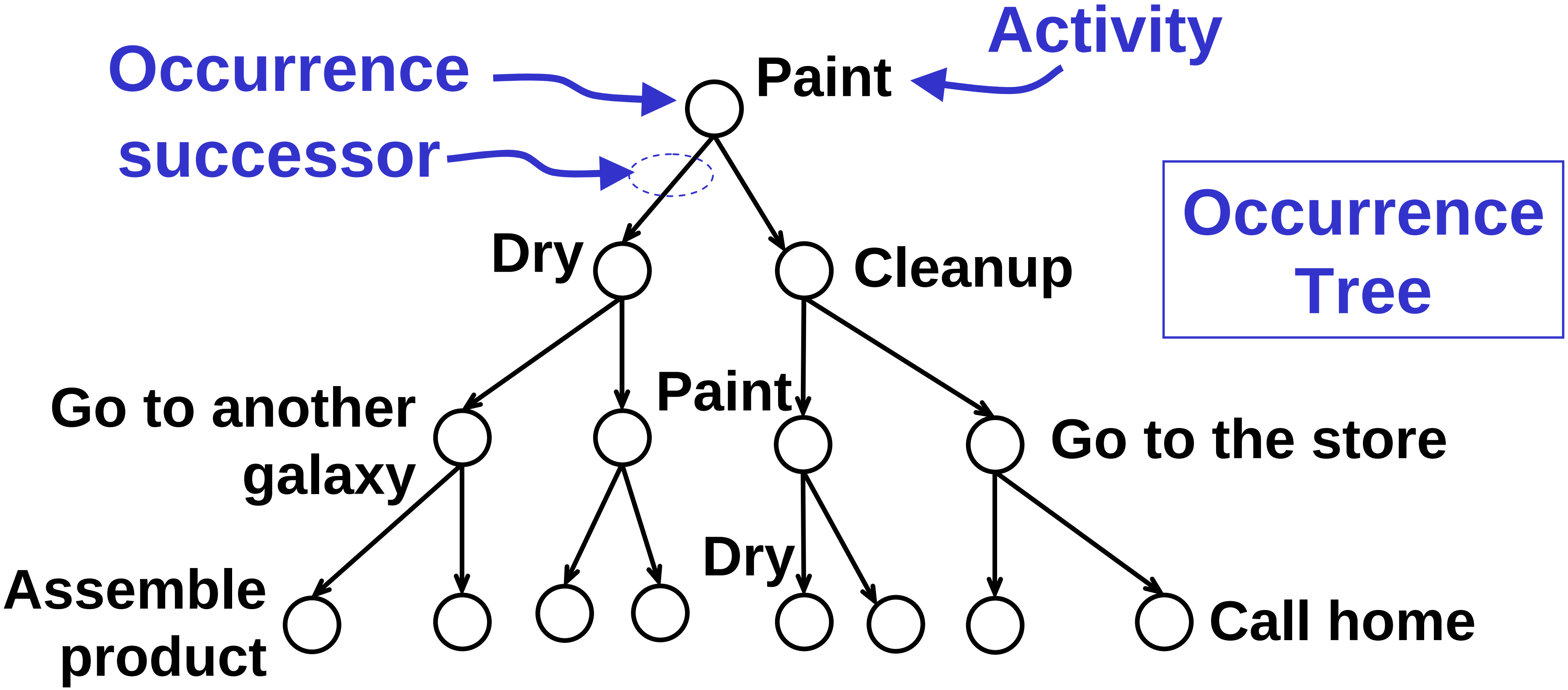
Process Specification Language

- An execution constraint language defined in first order (CLIF).
- ISO 18629 Full International Standard.
- Based on long period of research starting with situation calculus
- Applied to:
  - scheduling, process modeling, process planning, production planning, simulation, project management, workflow, business process reengineering, vehicle navigation, semantic interoperability.
- More information at http://www.nist.gov/psl.

Terminology Mapping

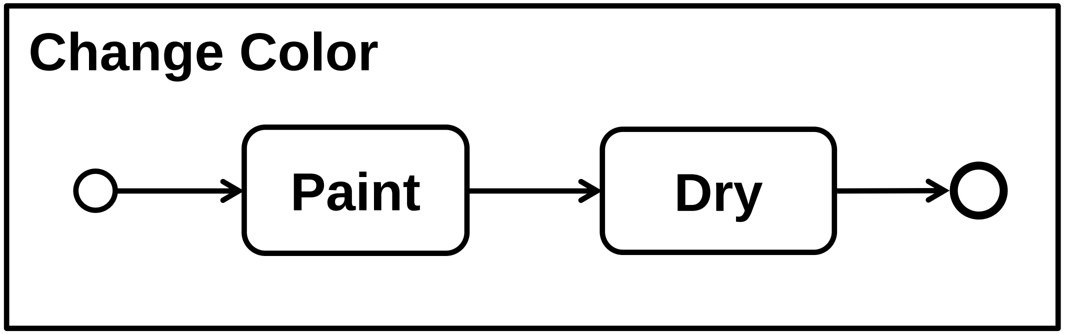
This presentation	PSL
Execution	(complex) occurrence
Process (M2)	activity
happensBefore	earlier (successor for "immediately before")
happensDuring (supertype of BPDM M1 happening parts)	subactivity_occurrence
BPDM Succession	min_precedes
Flow edge (BPDM ImmediateSuccession)	next_subocc

Anything Can Happen



- Tree of all possible execution sequences over entire world, including
  - not physically possible.
  - not specified by the user.
- Not stored anywhere, just referred to by constraints.

Processes in PSL

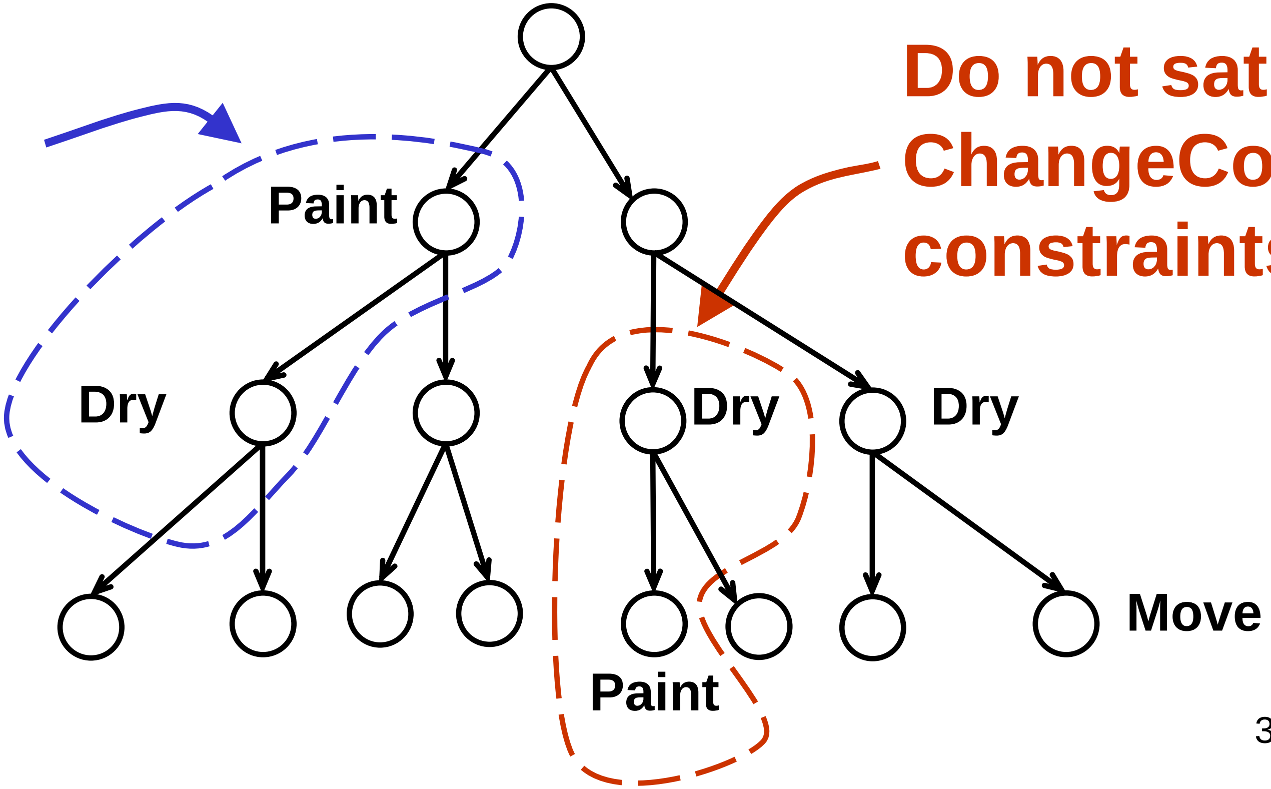


```
(forall (?occChangeColor)
  (implies
    (occurrence_of ?occChangeColor ChangeColor)
    (exists (?occPaint ?occDry)
      (and (occurrence_of ?occPaint Paint)
           (occurrence_of ?occDry Dry)
           (subactivity_occurrence ?occPaint ?occChangeColor)
           (subactivity_occurrence ?occDry ?occChangeColor)
           (min_precedes ?occPaint ?occDry
                         ChangeColor)))))
```

Processes in PSL

- **Portions of the occurrence tree (complex occurrences) will satisfy the constraints or not.**

Satisfy ChangeColor constraints.

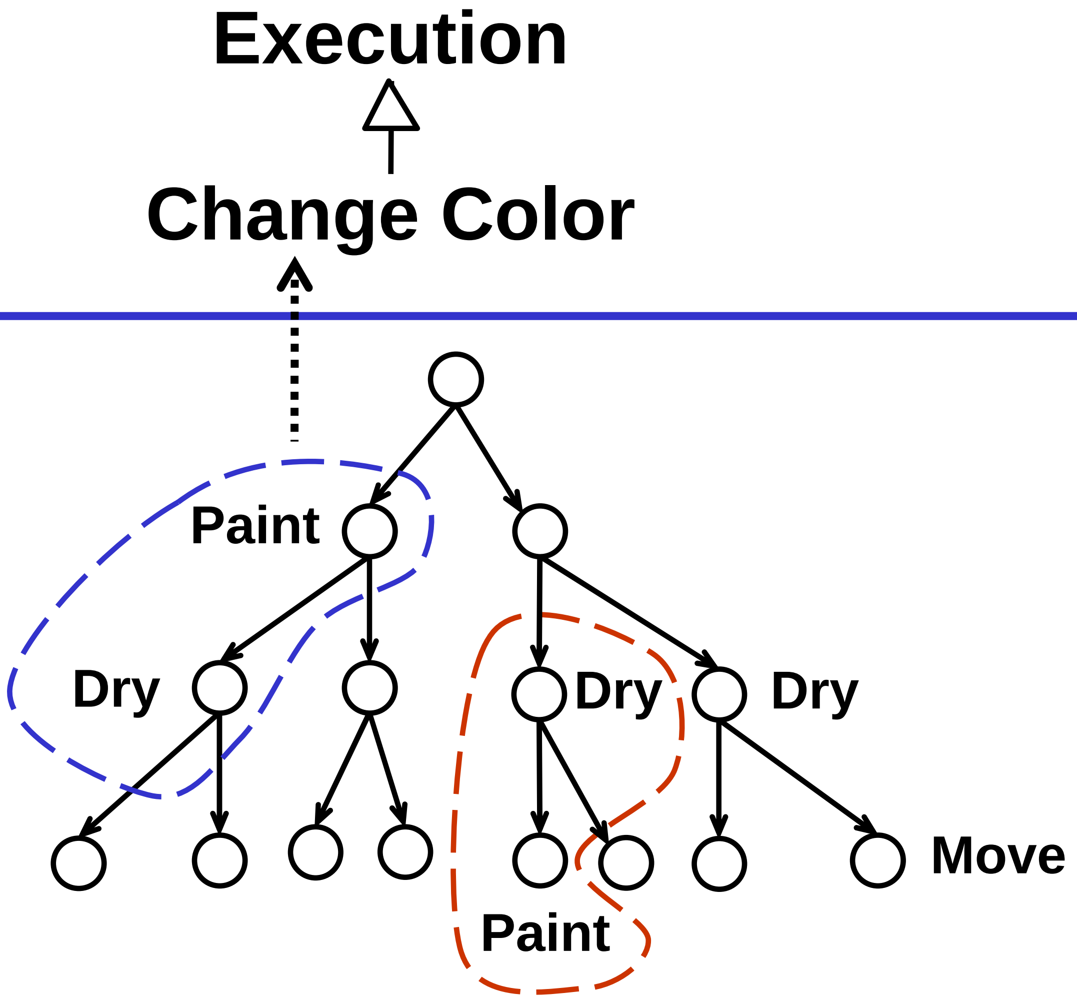


Do not satisfy ChangeColor constraints.

Process Modeling and PSL

Model (M1)

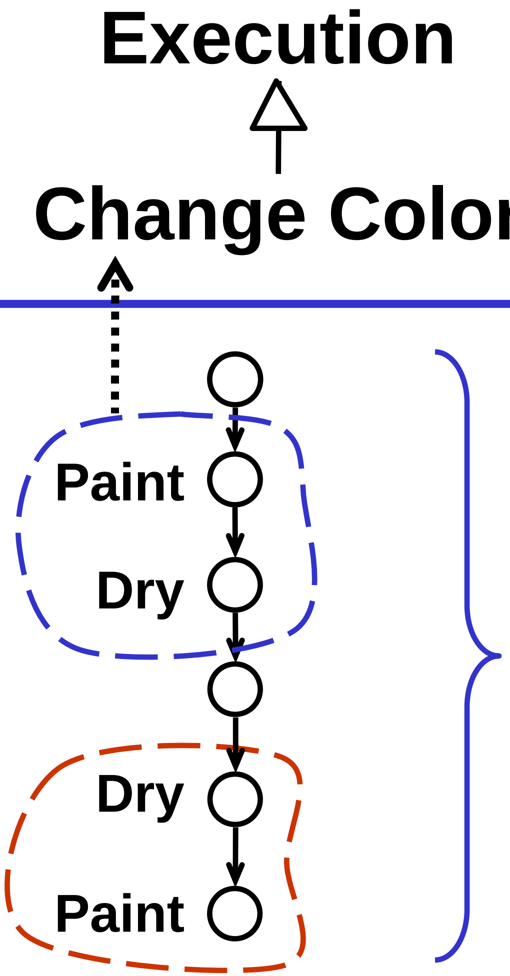
PSL Complex Occurrences



- **PSL complex occurrences "satisfy" process models (or not).**

Process Modeling and PSL

Model
(M1)



Execution

Change Color

Paint

Dry

Dry

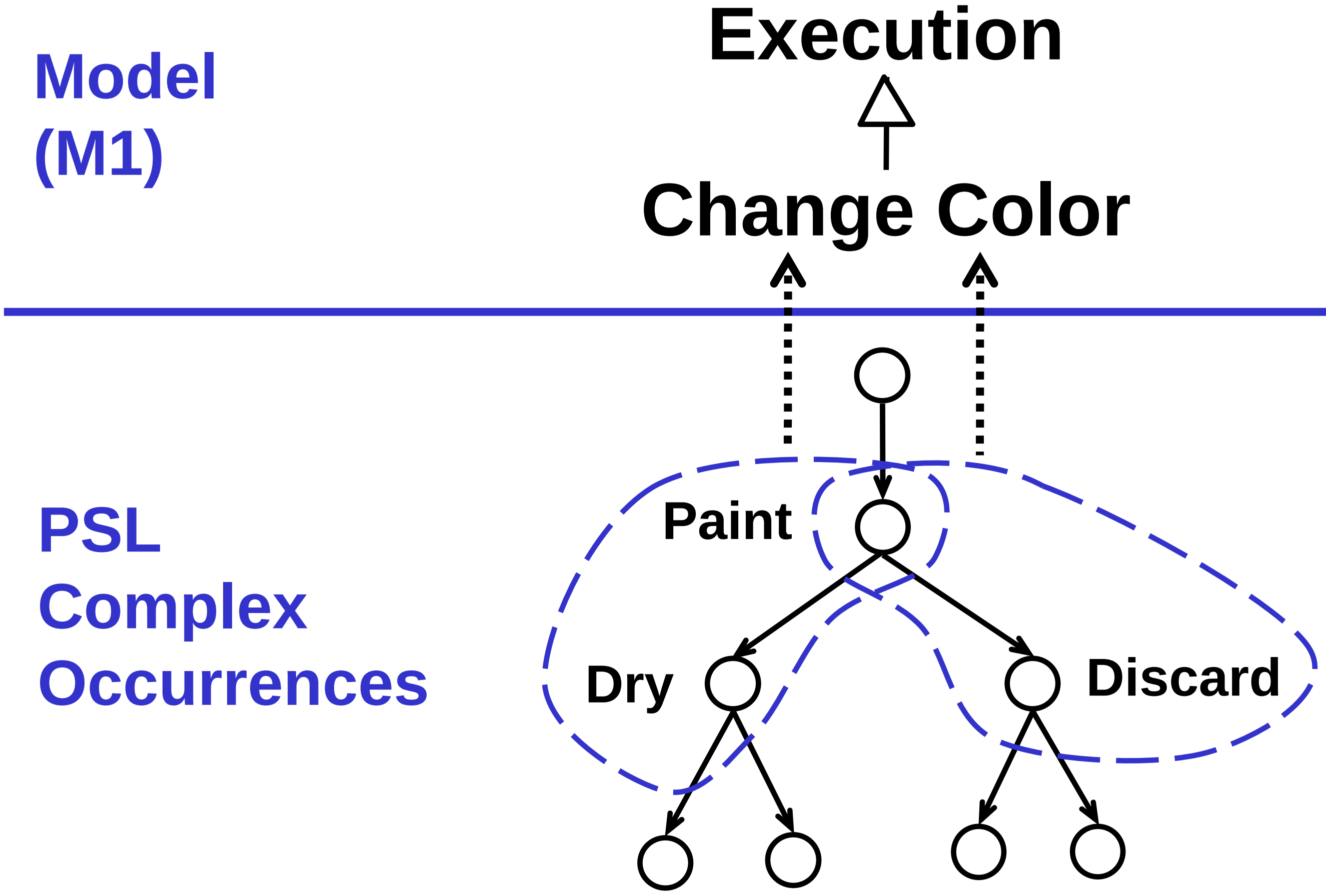
Paint

PSL
Complex
Occurrences

Single branch
occurrence tree

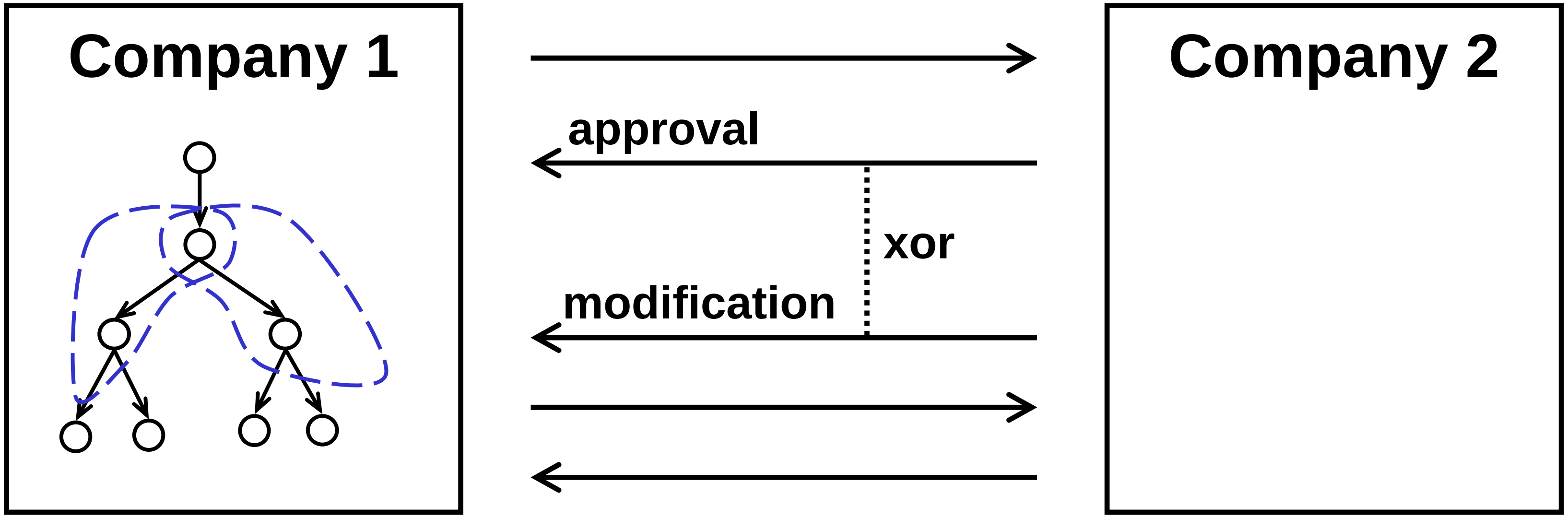
- **Without possibilities.**

Process Modeling and PSL



- **Required nondetermism, due to uncertainty about effects.**

Process Modeling and PSL



Interaction is a process (execution constraint)

- Required nondetermism applicable to service-level agreements, and choreography generally.
- Expressible in single-branch occurrence tree?

Process Modeling and PSL

- Translation of process models available:
  - incremental (fragments to small axioms)
  - handles loops, unstructured flows, input/output via parameters and messages, reaction to changes.
  - http://www.mel.nist.gov/msidlibrary/doc/NISTIR_7310.pdf
- U Toronto working on another approach.
- Best to combine the above.

Summary

- Models are shorthands for commonly used constraints on “instances” of the model.
- Process instances are executions.
- Extend metalanguages to incorporate executions into process modeling languages (BPDM) http://www.omg.org/cgi-bin/doc?dtc/07-11-01.
- Integrate PSL with process models by:
  - Constraint language for BPDM.
    - PSL complex occurrences as “instances”.
  - Direct translation from models to PSL constraints.