The Process Specification Language (PSL): Theories and Applications

Michael Grüninger and
Christopher Menzel

Journal Club Presentation

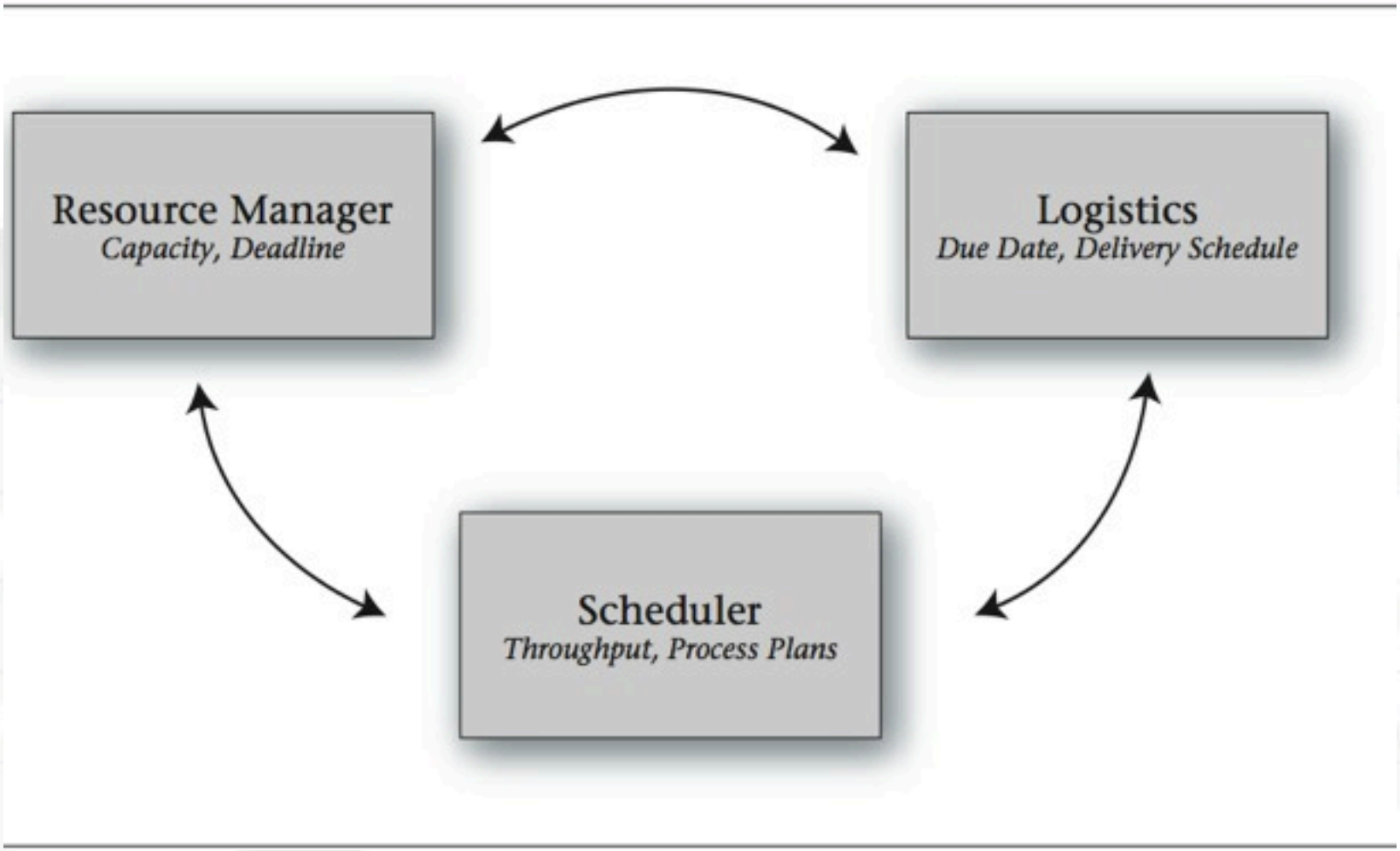
Eric Rozell, Tetherless World Constellation

Outline

- Overview
- Architecture
- Design Principles
- PSL in Action
- Significance for AST
- Take-aways

Overview

- Application integration is difficult
  - Different terminologies
  - Different semantics
- ***Goal:*** seamlessly exchange information between applications
- ***Naïve Approach:*** point-to-point translation
- ***PSL Approach:*** unambiguously specify terminology that define applications

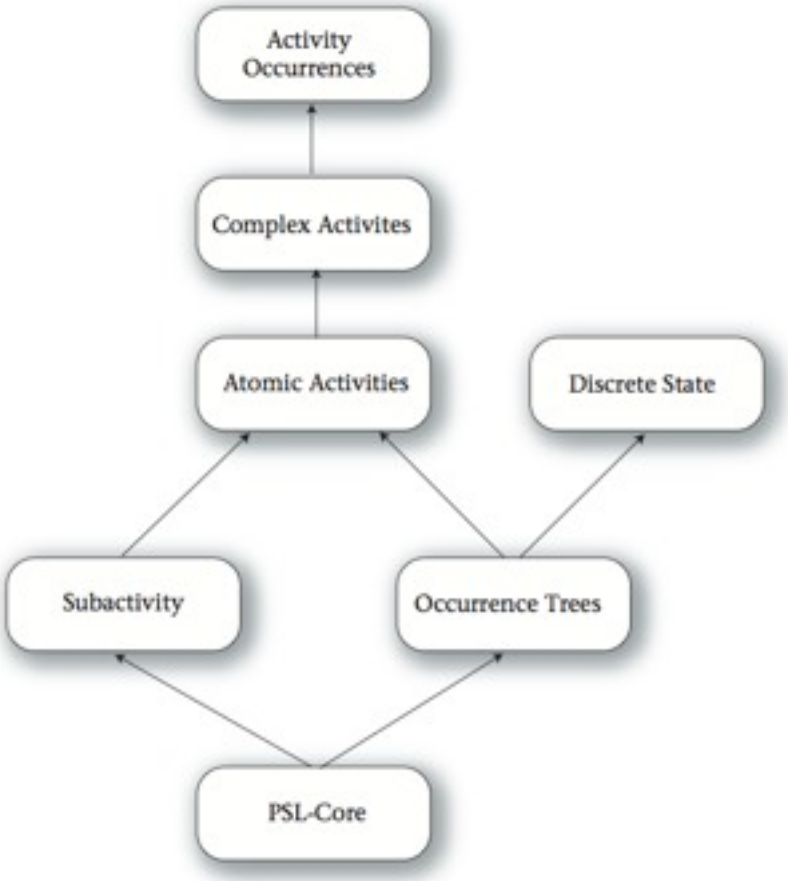

Resource Manager
Capacity, Deadline
Logistics
Due Date, Delivery Schedule
Scheduler
Throughput, Process Plans

Architecture

- PSL-Core
- PSL extensions
  - Core extensions: axiomatize primitives (i.e., adds expressivity)
  - Definitional extensions: uses only terminology from core theories (i.e., no expressivity added)
- Axioms are first-order sentences in Knowledge Interchange Format (KIF)

Architecture

- PSL-Core:
  - Four disjoint concepts:
    - Activities
    - Activity occurrences
    - Time points
    - Objects



Architecture

- "Outer Core"
  - Occurrence Trees
    - Isomorphic to situation trees in situation calculus
    - Can be "pruned" with poss relations
  - Discrete States
    - Adds notion of *state*
    - i.e., preconditions and effects for occurrences
  - Subactivities
    - Adds notion of discrete partial ordering of occurrences (no relation to Activities)

Architecture

- “Outer Core” cont’d
  - Atomic Activities
    - Like atomic instructions, allows aggregation of concurrent activities
    - e.g., Compare-and-Swap
  - Complex Activities
    - Defines relations between activities and subactivities
    - Occurrence of complex activity is a subtree of the Occurrence Tree
  - Activity Occurrences
    - Supports arbitrarily complex subactivities

Architecture

- Other Core Theories
  - Subactivity occurrence ordering, iterated occurrence ordering, duration, resource requirements

Definitional Extensions	Core Theories	Example Concepts
Activity Extensions	Complex Activities	Deterministic/Nondeterministic Activities
		Concurrent Activities
		Partially Ordered Activities
Temporal and State Extensions	Complex Activities	Preconditions
	Discrete States	Effects
		Conditional Activities
		Triggered Activities
Activity Ordering and Duration Extensions	Subactivity Occurrence Ordering	Complex Sequences and Branching
	Iterated Occurrence Ordering	Iterated Activities
	Duration	Duration-based Constraints
Resource Role Extensions	Resource Requirements	Reusable, Consumable, Renewable, Deteriorating Resources

Design Principles

- Supporting Interoperability
  - Interoperability Hypothesis
  - Restrict domain to first-order logic
  - Can occur where two applications have isomorphic models
- The Ontological Stance
  - Conformance Hypothesis
  - Authors assume that any application can be modeled in first-order logic
  - Also assumes PSL will be expressive enough to axiomatize any application

Design Principles

- Characterization of Models
  - Definability Criterion
  - Intuitions and structures
  - “an empirical approach”
- The Role of Definitional Extensions
  - Core theories are defined with respect to “invariants”
  - Classification Criterion
  - Definitional Extension Criterion

PSL in Action

- Activity Role Declarations

  (define-activity-role
  :id <number>
  :name <string>
  :successors <number>*
  :preconditions <PSL sentence>*
  :postconditions <PSL sentence>*)

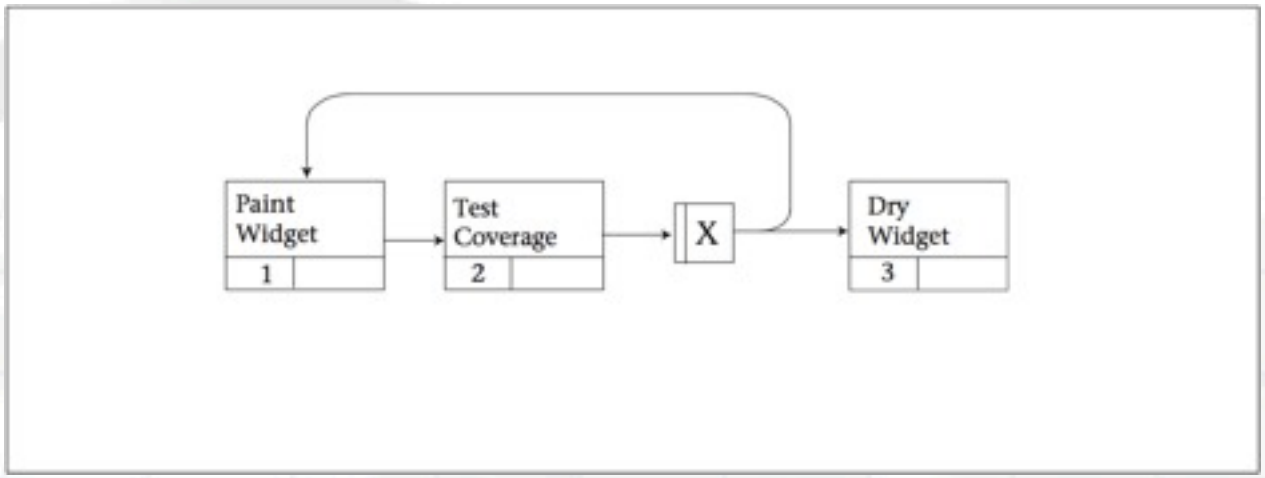
- Object Declarations

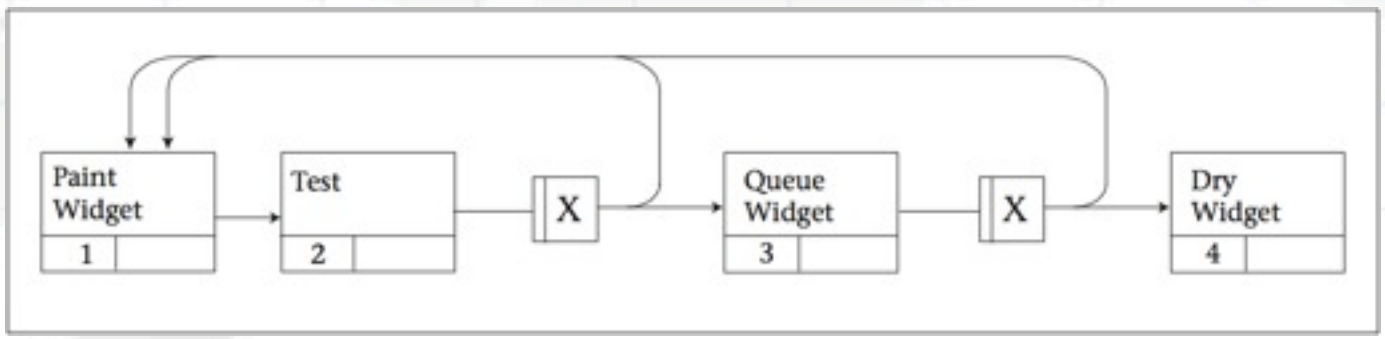
  (define-object
  :name <KIF constant>
  :constraints <PSL sentence>*)

- Parameter Declarations

  (define-parameter
  :variable <KIF variable>

PSL in Action





PSL in Action

```
(define-object
  :name widget
  :constraints (Widget widget))

(define-object
  :name painter
  :constraints (Paint_Sprayer painter))

(define-object
  :name oven
  :constraints (Oven oven))

(define-activity-role
  :id Act-1
  :name Paint_Widget
  :successors 2
  :preconditions
    (or (not (Painted widget (beginof ?occ)))
        (not (Adequate (Paint_Coverage widget (beginof ?occ)))))
  :postconditions
    (Painted widget (endof ?occ)))

(define-activity-role
  :id Act-2
  :name Test_Coverage
  :successors 1 3
  :preconditions (Painted widget (beginof ?occ))
  :postconditions (Adequate (Paint_Coverage widget) (endof ?occ)))

(define-activity-role
  :id Act-3
  :name Dry_Widget
  :successors
  :preconditions (Adequate (Paint_Coverage widget) (beginof ?occ))
  :postconditions (Dry widget (endof ?occ))
```

```
(define-parameter
  :variable ?w
  :constraints (Widget ?w))

(define-activity-role
  :id Act-1
  :name Paint_Widget
  :successors 2
  :preconditions
    (or (not (Painted ?w (beginof ?occ)))
        (not (Adequate (Paint_Coverage ?w (beginof ?occ)))))
  :postconditions
    (Painted widget (endof ?occ)))

(define-activity-role
  :id Act-2
  :name Test_Coverage
  :successors 1 3
  :preconditions (Painted ?w (beginof ?occ))
  :postconditions (Adequate (Paint_Coverage widget) (endof ?occ)))

(define-activity-role
  :id Act-3
  :name Queue_Widget
  :successors 1 4
  :preconditions
    (Adequate (Paint_Coverage ?w (beginof ?occ)))
  :postconditions
    (Painted widget (endof ?occ)))

(define-activity-role
  :id Act-4
  :name Dry_Widget
  :successors
  :preconditions (Adequate (Paint_Coverage ?w) (beginof ?occ))
  :postconditions (Dry ?w (endof ?occ))
```

Significance for AST

- Health IT Infrastructure
  - Service-oriented architecture dismissed by PCAST
  - Definitional extensions to a PSL-based SOA could enable infrastructure
- Water Quality Portal
  - Express details of processes involved in water quality tests (i.e., provenance)
- S2S
  - Process models enable web service integration and composition

Take-aways

- Provides an ontology for:
  - Describing various activity types (i.e., complex, atomic, sub-)
  - Describing system states and activity occurrences (“instances” of activities)
- Rigorous semantics captured in core axioms
- Supports “empirical” approach (i.e., extensions can be mutually inconsistent)

Questions?

- Thank you!