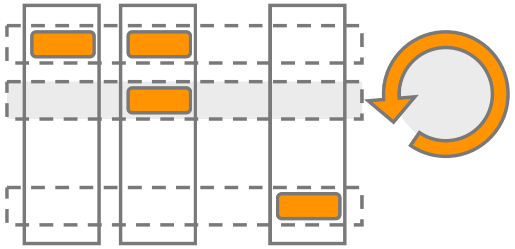
Foundations of Aspect-Oriented Languages (FOAL)
Modularity 2015

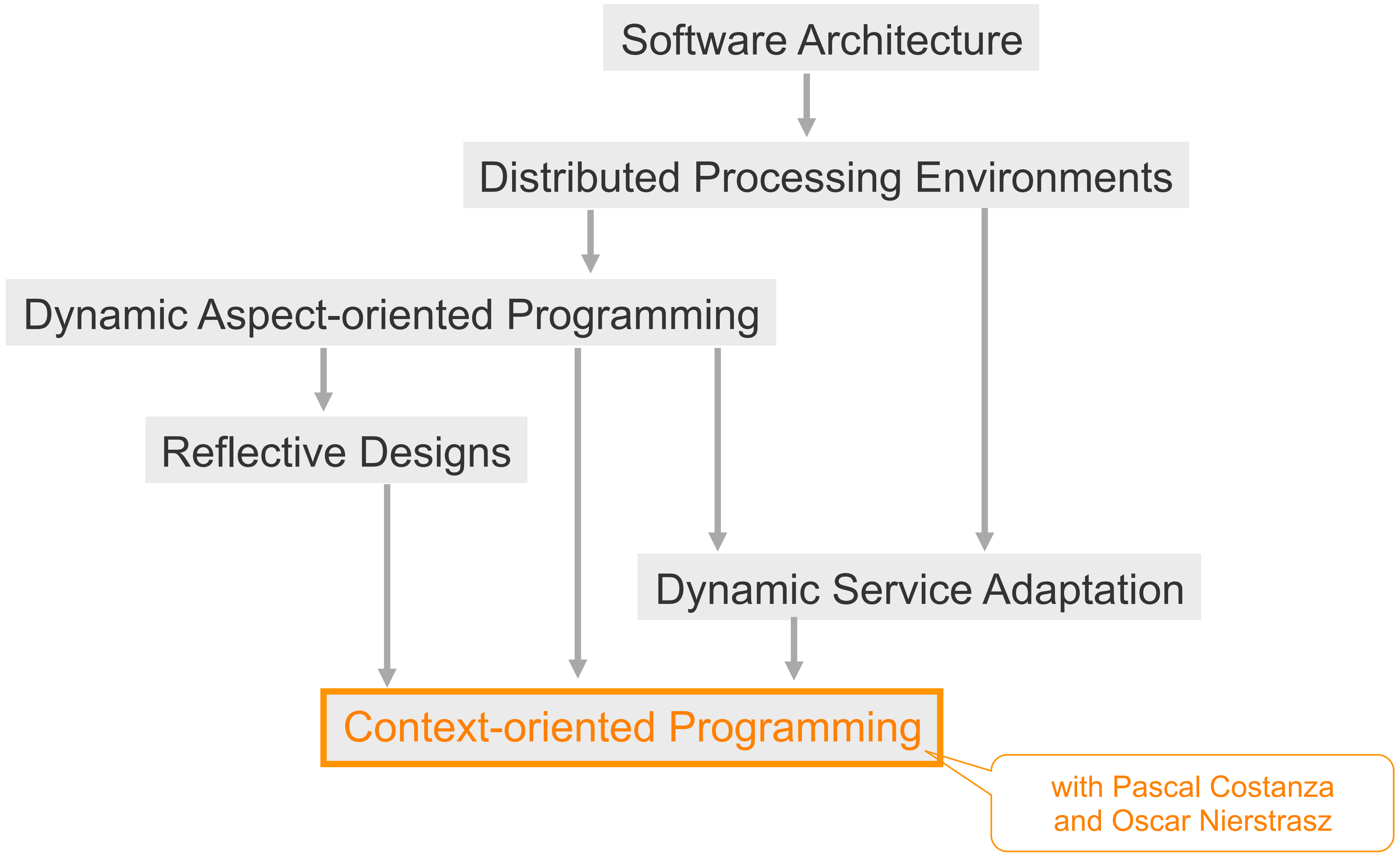


Recent Developments in Context-oriented Programming
(at HPI)

Robert Hirschfeld
Hasso Plattner Institute
University of Potsdam
Germany
http://www.hpi.de/swa/

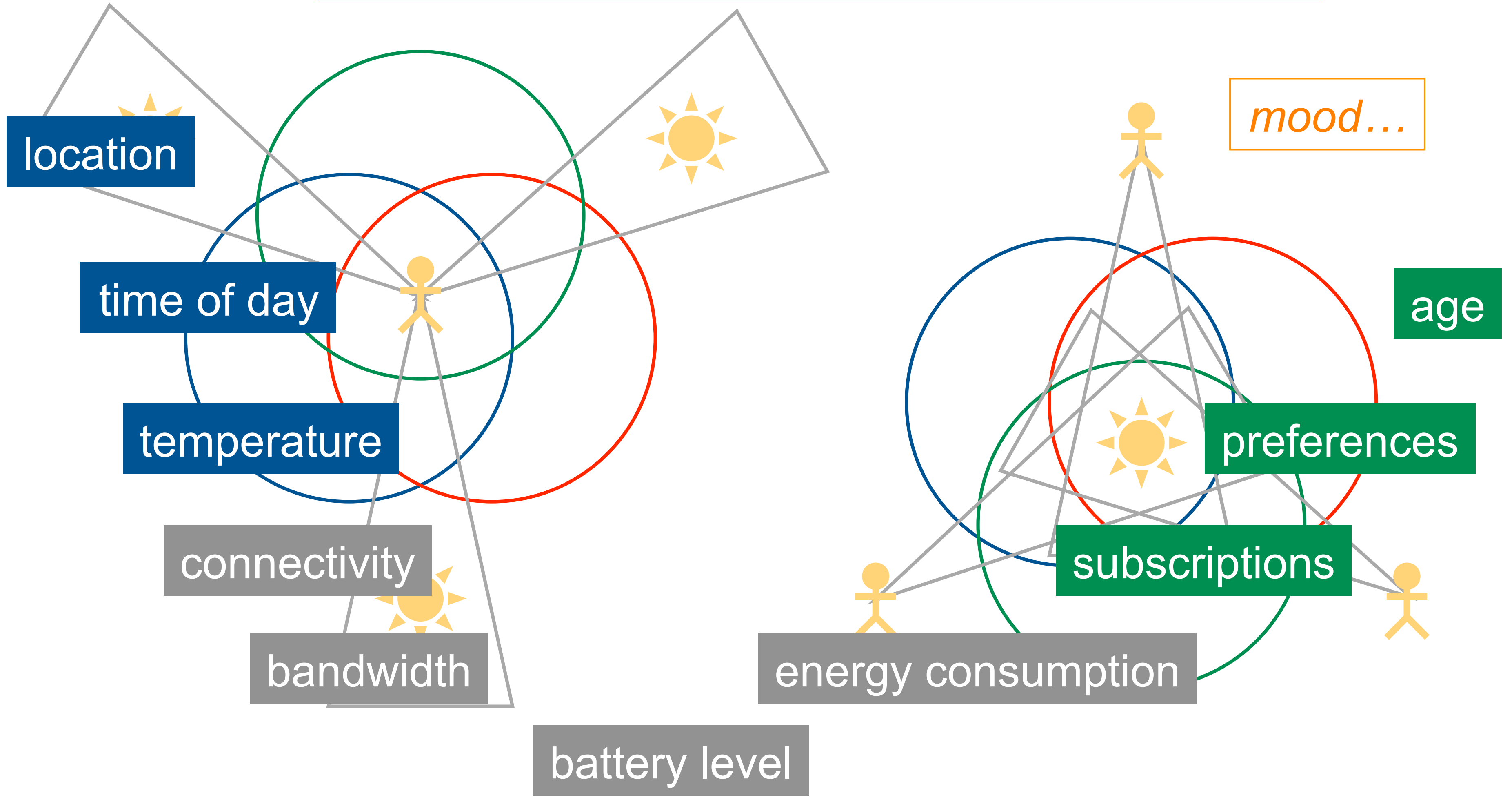
Colorado State University, Fort Collins, Colorado
2015-03-16

Some History…

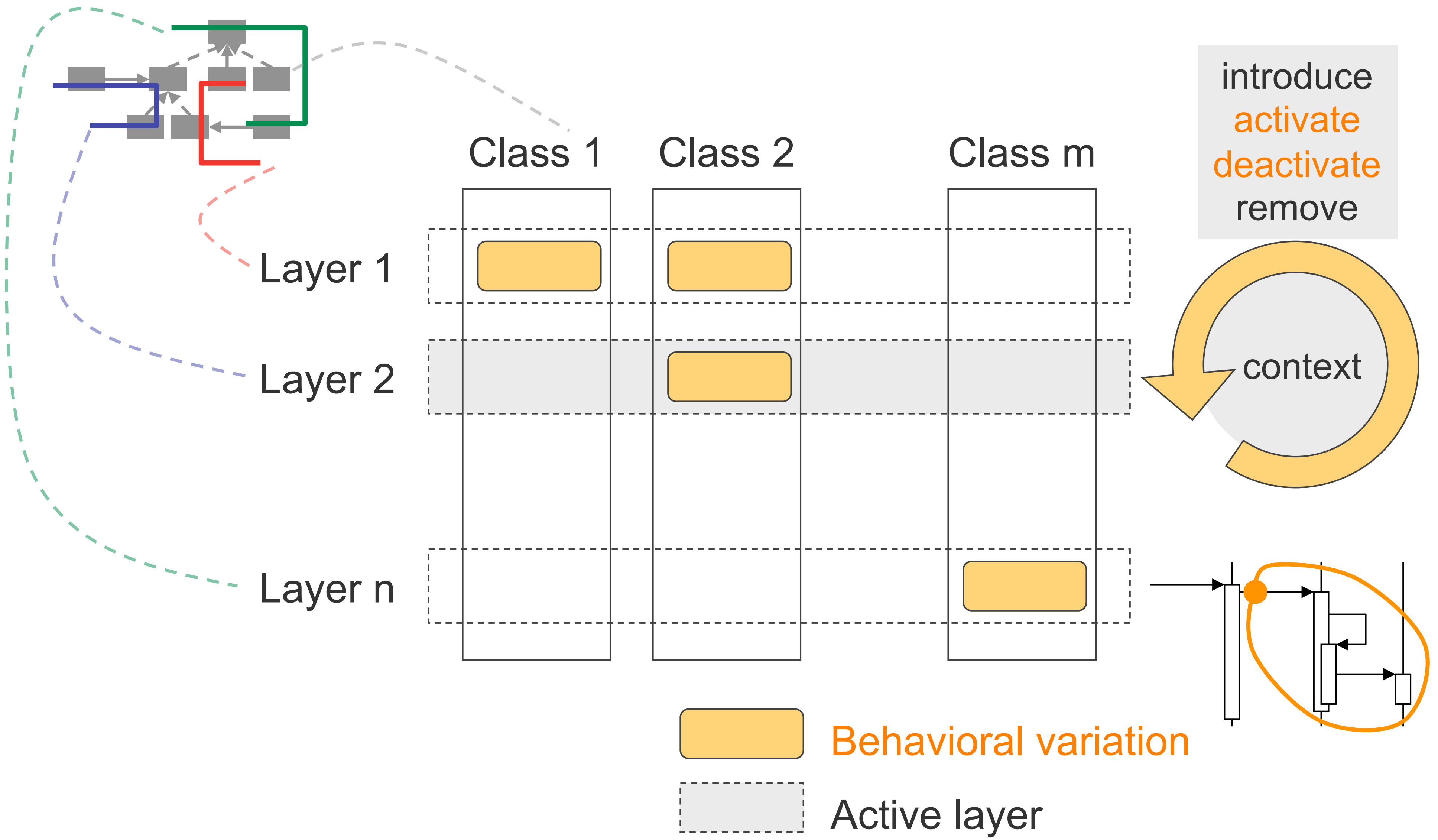

Software Architecture
Distributed Processing Environments
Dynamic Aspect-oriented Programming
Reflective Designs
Dynamic Service Adaptation
Context-oriented Programming
with Pascal Costanza and Oscar Nierstrasz

Context

context = everything computationally accessible



COP Basics Overview

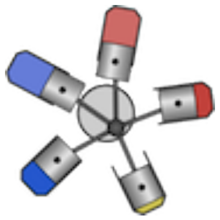
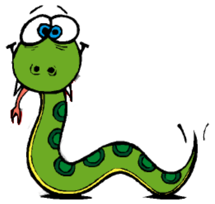


AOP, FOP, and COP

	AOP	FOP	COP
Inverse dependencies	●		
1:n relationships	●		
Layers		●	●
Dynamic activation			●
Scoping	●		●

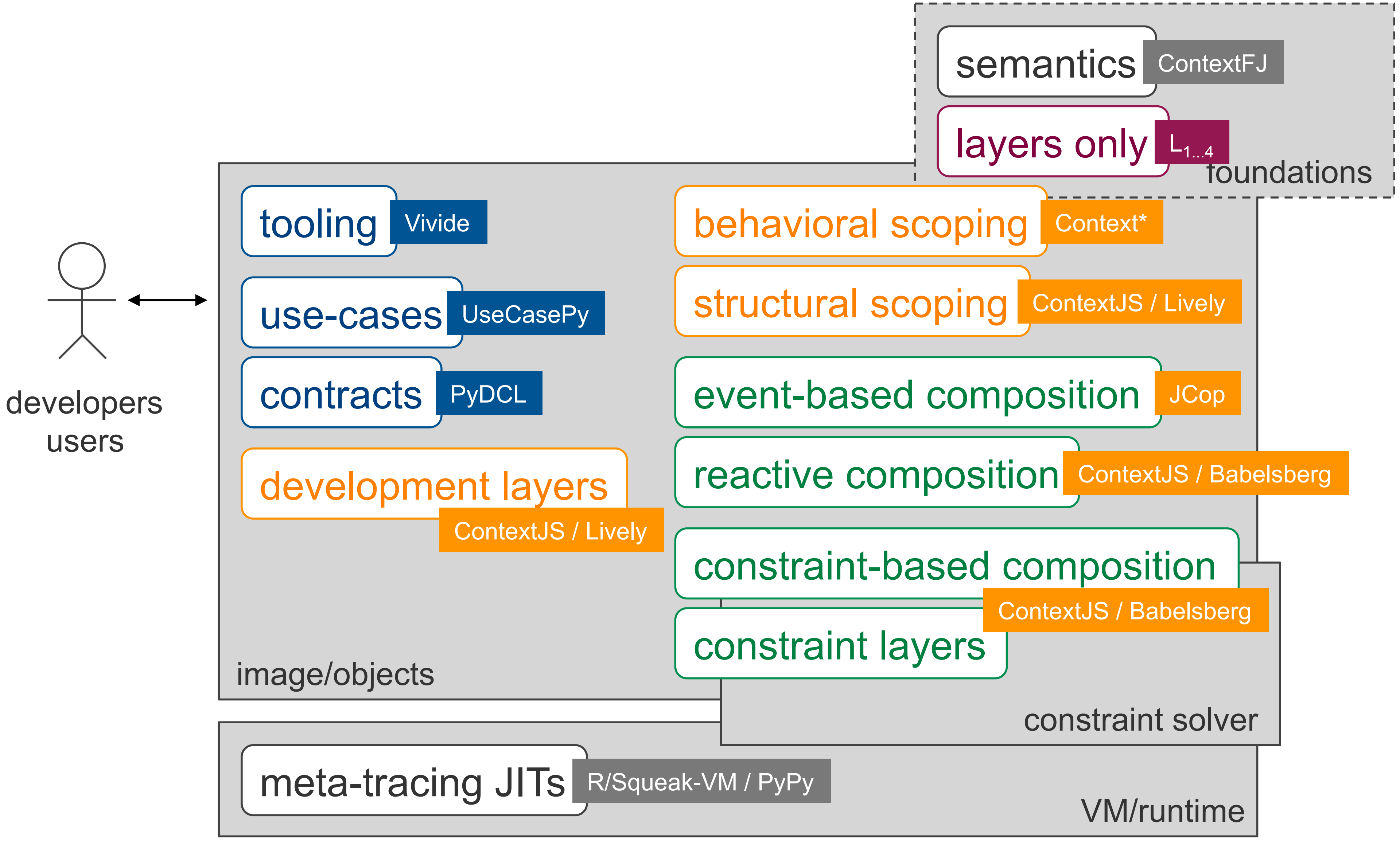
COP Extensions (Some…)

- ContextS
- ContextS2
- ContextJS
- JCop (ContextJ)
- ContextPy
- PyDCL
- UseCasePy
- PyContext
- ContextR
- ContextG
- ContextAmber
- $L_{1\ldots4}$

- ContextL
- ContextScheme
- ContextJ*
- ContextErlang
- EventCJ
- Lambic
- Ambience
- COP.JS
- delMDSCO/cj
- Phenomenal Gem
- Subjective-C
- Context Petri Nets

Recent COP Developments at HPI



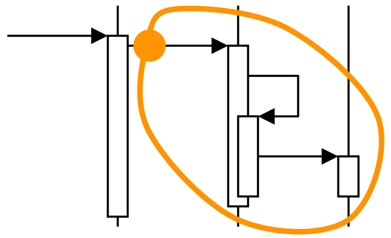
Behavioral Variations

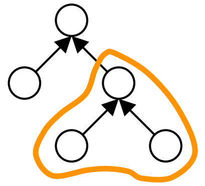
- Behavioral (dynamic) scoping
  - Dynamic extent of execution
  - Almost all COP extensions
- Structural (topological) scoping
  - ContextJS
  - Development layers
- Open implementation for scoping strategies
  - Allows for domain-specific scoping
  - Mainly applied to UI framework structures
    - Lively: Morphic
    - Webwerkstatt : Parts

behavioral scoping Context*

structural scoping ContextJS / Lively

development layers ContextJS / Lively





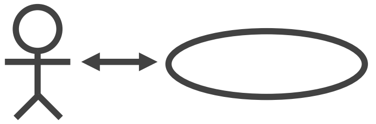
Development Support

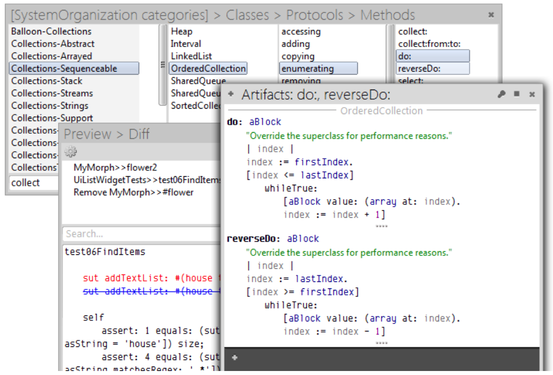
- More applied → more useful
- In PL work tool support often neglected
  - Usually too expensive, especially early…
    → Need for explorative tool building support
    - Vivide
- Crosscutting nature of layers lends itself nicely to crosscutting software engineering concerns
  - Explicit use-cases representation
    - UseCasePy
  - Dynamic contract layers
    - PyDCL

tooling Vivide

use-cases UseCasePy

contracts PyDCL





MyMorph>>flower2
UiListWidgetTests>>test06FindItem
Remove MyMorph>>#flower

```
test06FindItems

   sut addTextList: #(house
   ~~sut addTextList: #(house~~

   self
      assert: 1 equals: (sut
asString = 'house']) size;
      assert: 4 equals: (sut
```

```
do: aBlock
   "Override the superclass for performance reasons."
   | index |
   index := firstIndex.
   [index <= lastIndex]
      whileTrue:
         [aBlock value: (array at: index).
         index := index + 1]

reverseDo: aBlock
   "Override the superclass for performance reasons."
   | index |
   index := lastIndex.
   [index >= firstIndex]
      whileTrue:
         [aBlock value: (array at: index).
         index := index - 1]
```

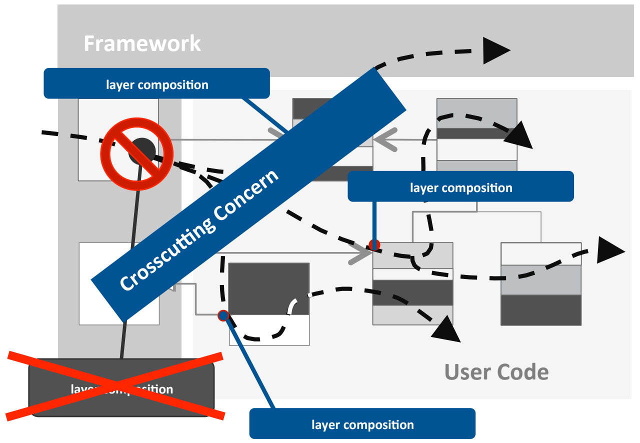
Reactive Approaches

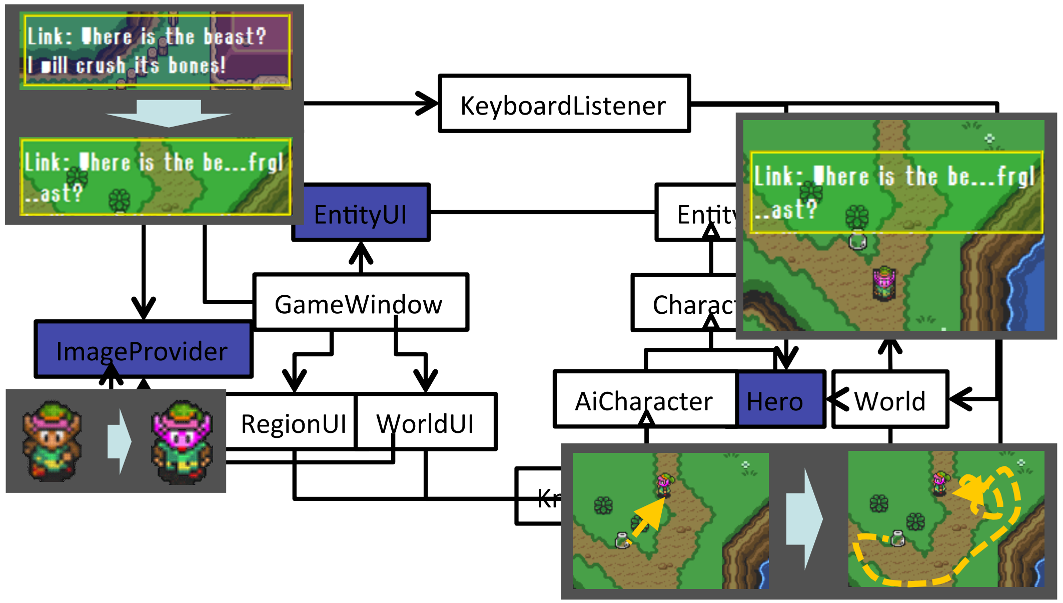
event-based composition — JCop

reactive composition — ContextJS / Babelsberg

constraint-based composition — ContextJS / Babelsberg

constraint layers




Link: Where is the beast? I will crush its bones!
Link: Where is the be...frgl ..ast?
KeyboardListener
EntityUI
Entity
GameWindow
Charac
ImageProvider
RegionUI
WorldUI
AiCharacter
Hero
World
Link: Where is the be...frgl ..ast?

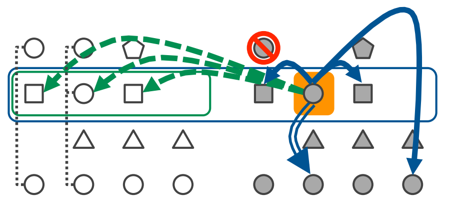
Foundations

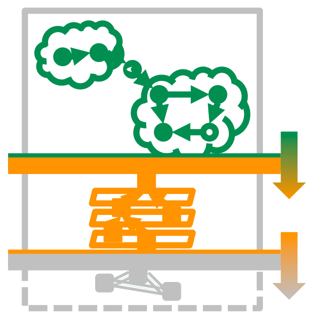
semantics ContextFJ

layers only $L_{1...4}$

meta-tracing JITs R/Squeak-VM / PyPy

- Semantics and types
  - ContextFJ
- Symmetry
  - No classes, only layers
  - No base system
    - $L_{1..4}$
- Sideways composition very expensive
  - Runtime support for optimizations
  - Meta-tracing JITs
    - R/Squeak-VM
  - Higher performance → more (meta-level) flexibility





$$\frac{PT(\mathtt{m},\mathtt{C},\mathtt{L}_0)\text{ undefined} \qquad mbody(\mathtt{m},\mathtt{C},\overline{\mathtt{L}}',\overline{\mathtt{L}}) = \overline{\mathtt{x}}.\mathtt{e}\ \mathtt{in}\ \mathtt{D},\overline{\mathtt{L}}''}{mbody(\mathtt{m},\mathtt{C},(\overline{\mathtt{L}}';\mathtt{L}_0),\overline{\mathtt{L}}) = \overline{\mathtt{x}}.\mathtt{e}\ \mathtt{in}\ \mathtt{D},\overline{\mathtt{L}}''}$$

Acknowledgements

Pascal Costanza, Hidehiko Masuhara, Atsushi Igarashi, Michael Haupt, Malte Appeltauer, Michael Perscheid, Bastian Steinert, Jens Lincke, Marcel Taeumel, Tobias Pape, Tim Felgentreff, Robert Krahn, Carl Friedrich Bolz, Marcel Weiher, Hans Schippers, Tim Molderez, Oscar Nierstrasz, Shigeru Chiba, Hiroaki Inoue, Tobias Rho, Stefan Udo Hanenberg, Dick Gabriel, Dave Thomas, Gilad Bracha, Alan Kay, Dan Ingalls, Alan Borning, Jeff Eastman, Christopher Schuster, Christian Schubert, Gregor Schmidt, Stefan Lehmann, Matthias Springer, …