geodes Research Day AND Symposium

Invited talk at
GEODES Research Day and Symposium
Montreal, 2022

Overview of our research in software modeling and simulation

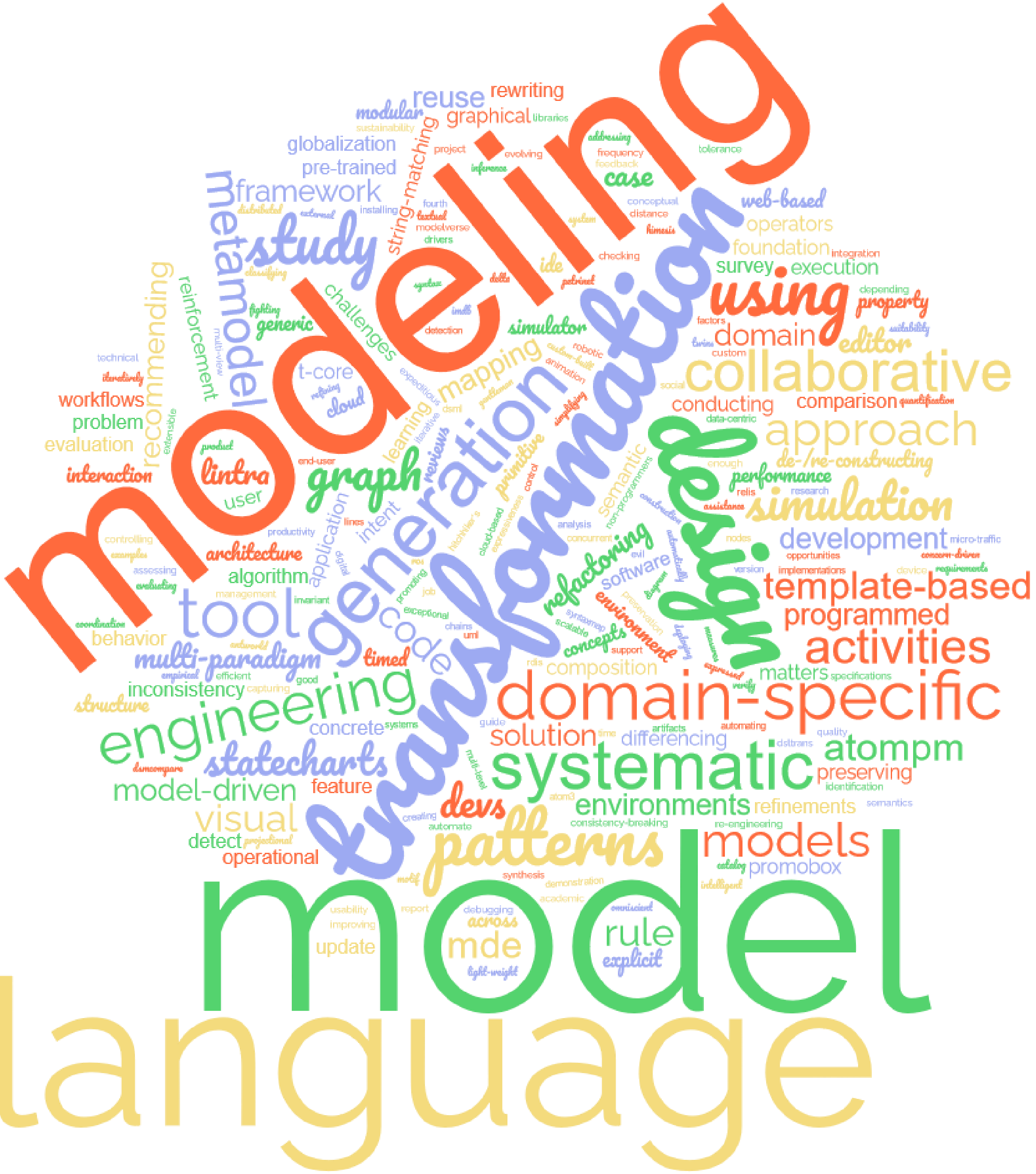
Prof. Eugene Syriani

Software Modeling & Simulation

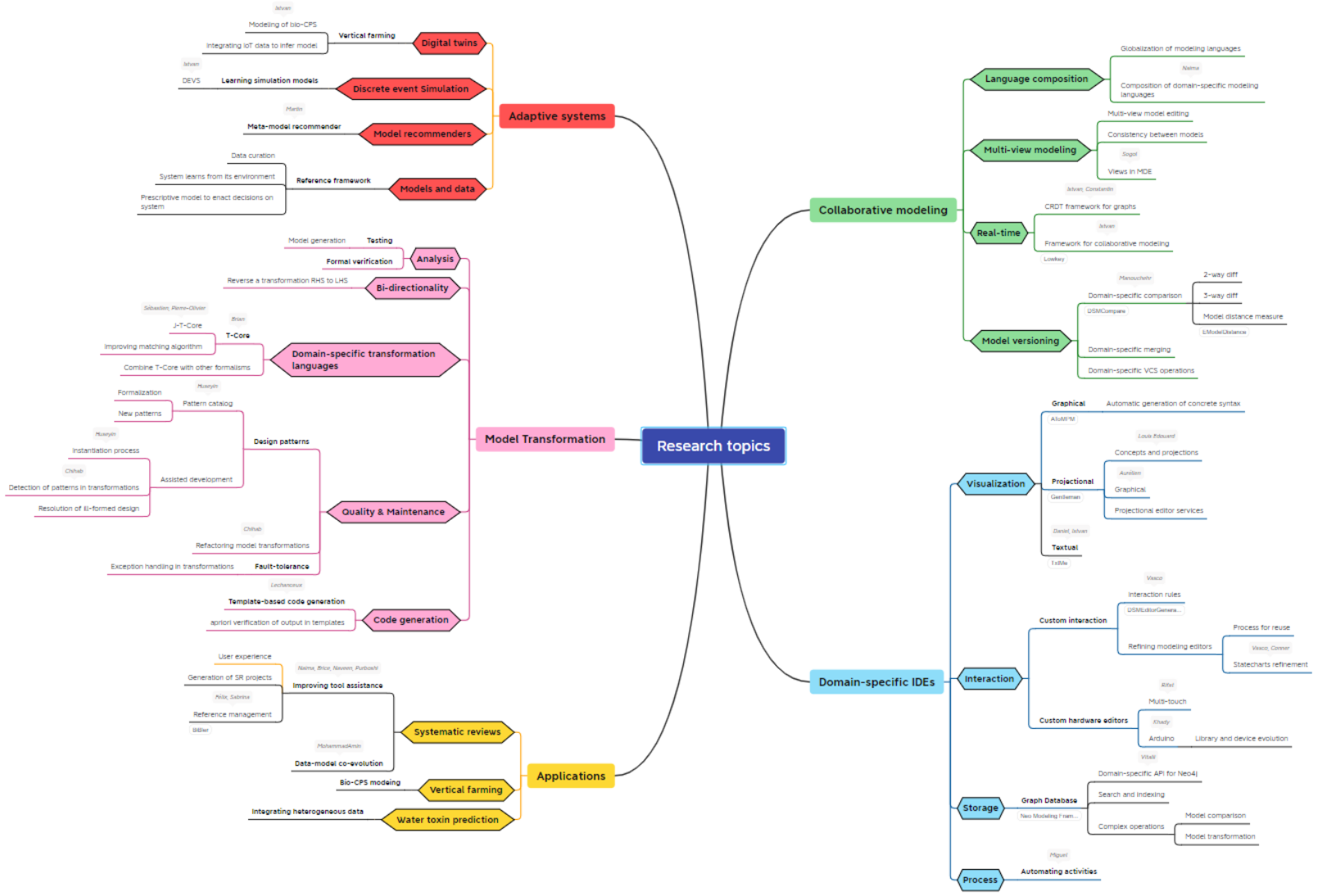
geodes

DIRO

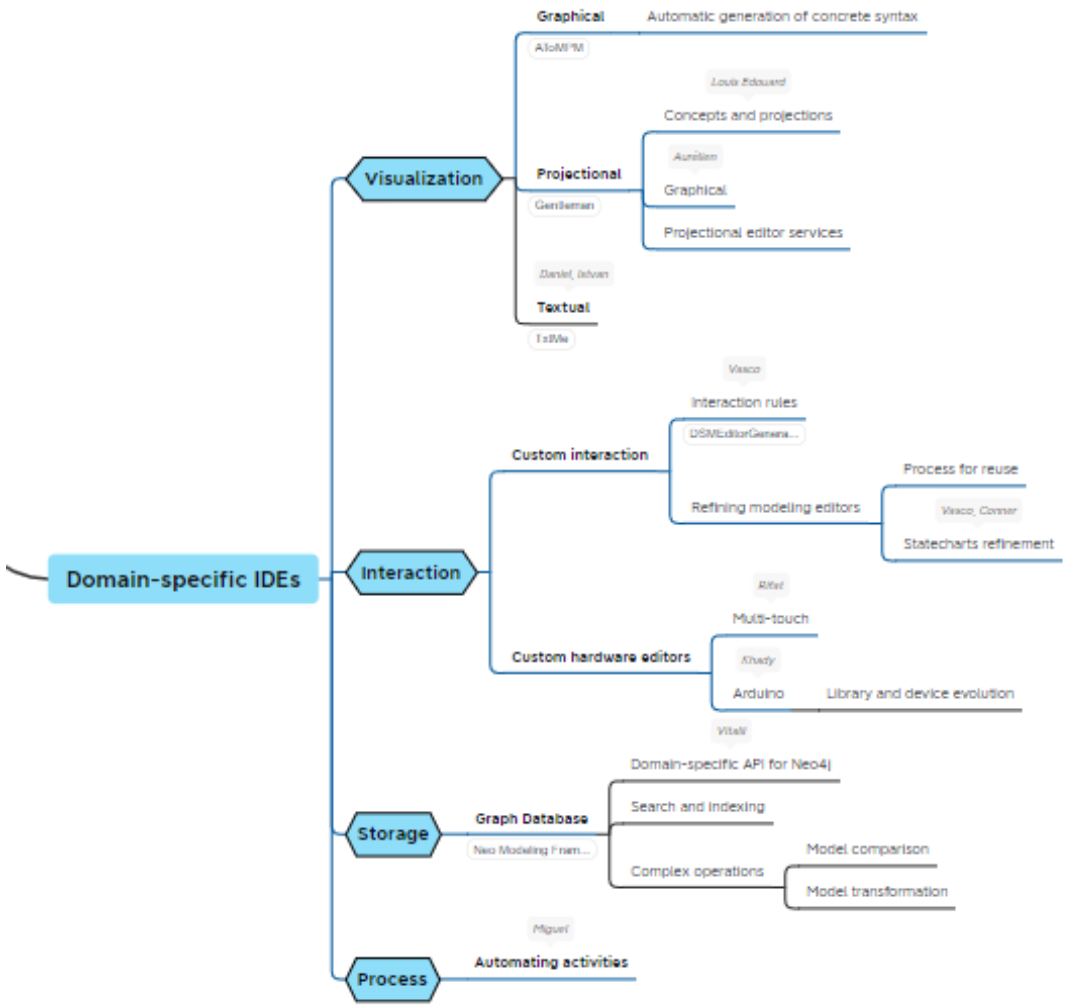
Université de Montréal


model
language
modeling
transformation
design
domain-specific
systematic
engineering
generation
collaborative
approach
patterns
tool
study
metamodel
graph
statecharts
simulation
template-based
activities
programmed
development
atompm
models
environments
refactoring
multi-paradigm
using
domain
recommending
reinforcement
framework
reuse
rewriting
graphical
modular
globalization
pre-trained
string-matching
challenges
mapping
t-core
cloud
lintra
interaction
workflows
problem
evaluation
architecture
algorithm
user
intent
application
code
timed
behavior
inconsistency
structure
model-driven
visual
detect
operational
feature
concrete
solution
differencing
composition
matters
preserving
refinements
promobox
rule
mde
update
explicit
across
devs
conducting
comparison
de-/re-constructing
performance
survey
execution
foundation
operators
web-based
case
property
editor
simulator
generic
semantic
software
concepts
environment
primitive
reviews
learning

Outline


Research topics
Adaptive systems
Digital twins
Vertical farming
Modeling of bio-CPS
Integrating IoT data to infer model
Discrete event Simulation
Learning simulation models
DEVS
Model recommenders
Meta-model recommender
Models and data
Reference framework
Data curation
System learns from its environment
Prescriptive model to enact decisions on system
Model Transformation
Analysis
Model generation
Testing
Formal verification
Bi-directionality
Reverse a transformation RHS to LHS
Domain-specific transformation languages
T-Core
J-T-Core
Improving matching algorithm
Combine T-Core with other formalisms
Design patterns
Pattern catalog
Formalization
New patterns
Assisted development
Instantiation process
Detection of patterns in transformations
Resolution of ill-formed design
Quality & Maintenance
Refactoring model transformations
Fault-tolerance
Exception handling in transformations
Code generation
Template-based code generation
apriori verification of output in templates
Applications
Systematic reviews
Improving tool assistance
User experience
Generation of SR projects
Reference management
Data-model co-evolution
Vertical farming
Bio-CPS modeling
Water toxin prediction
Integrating heterogeneous data
Collaborative modeling
Language composition
Globalization of modeling languages
Composition of domain-specific modeling languages
Multi-view modeling
Multi-view model editing
Consistency between models
Views in MDE
Real-time
CRDT framework for graphs
Framework for collaborative modeling
Model versioning
Domain-specific comparison
2-way diff
3-way diff
Model distance measure
Domain-specific merging
Domain-specific VCS operations
Domain-specific IDEs
Visualization
Graphical
Automatic generation of concrete syntax
Projectional
Concepts and projections
Graphical
Projectional editor services
Textual
Interaction
Custom interaction
Interaction rules
Refining modeling editors
Process for reuse
Statecharts refinement
Custom hardware editors
Multi-touch
Ardulno
Library and device evolution
Storage
Graph Database
Domain-specific API for Neo4j
Search and Indexing
Complex operations
Model comparison
Model transformation
Process
Automating activities

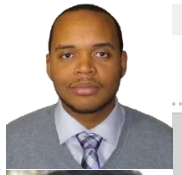
Outline

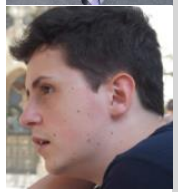

Domain-specific IDEs
Visualization
Graphical
Automatic generation of concrete syntax
AToMPM
Louis Edouard
Projectional
Concepts and projections
Aurelien
Graphical
Gentleman
Projectional editor services
Daniel, Istvan
Textual
TxtMe
Interaction
Vasco
Custom interaction
Interaction rules
DSMEditorGenera...
Refining modeling editors
Process for reuse
Vasco, Conner
Statecharts refinement
Rifel
Custom hardware editors
Multi-touch
Khady
Arduino
Library and device evolution
Vitali
Storage
Graph Database
Neo Modeling Fram...
Domain-specific API for Neo4j
Search and indexing
Complex operations
Model comparison
Model transformation
Miguel
Process
Automating activities

Domain-specific IDEs – Editors

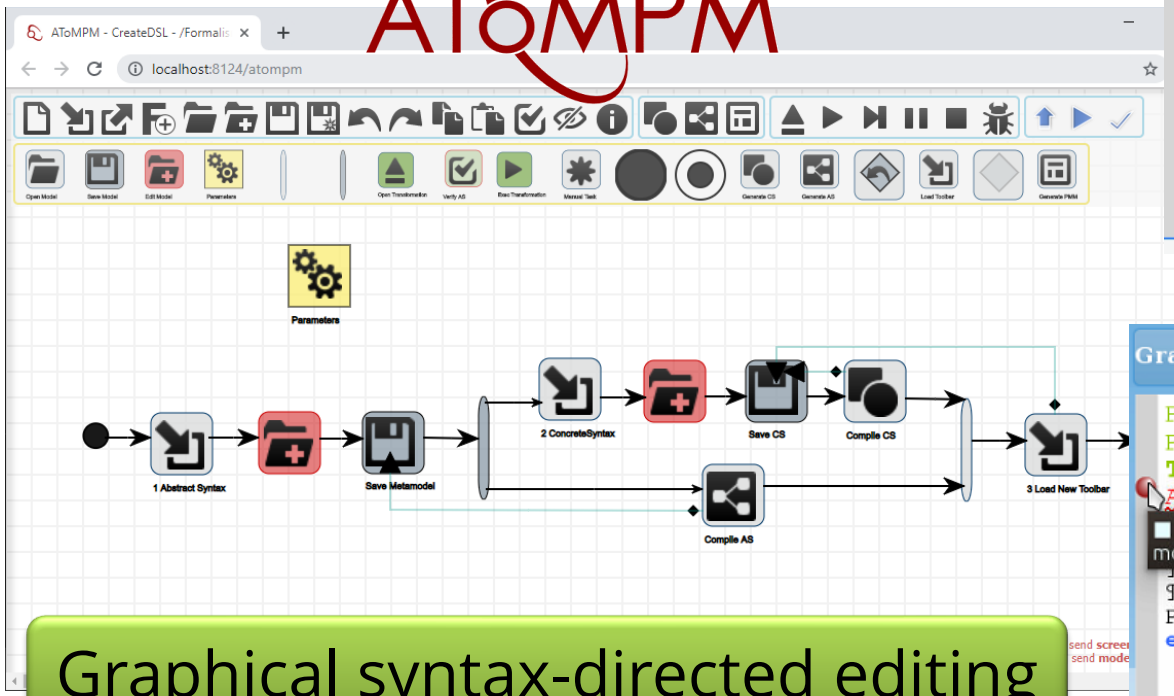
THE UNIVERSITY OF ALABAMA

McGill



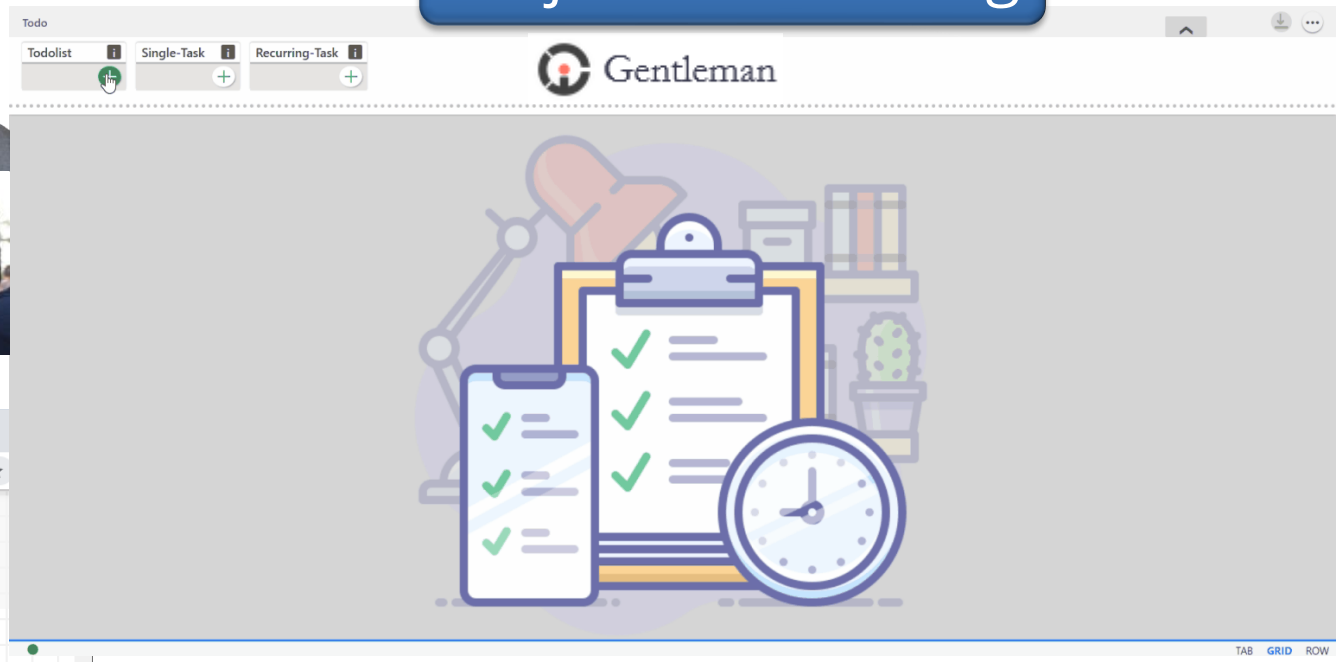


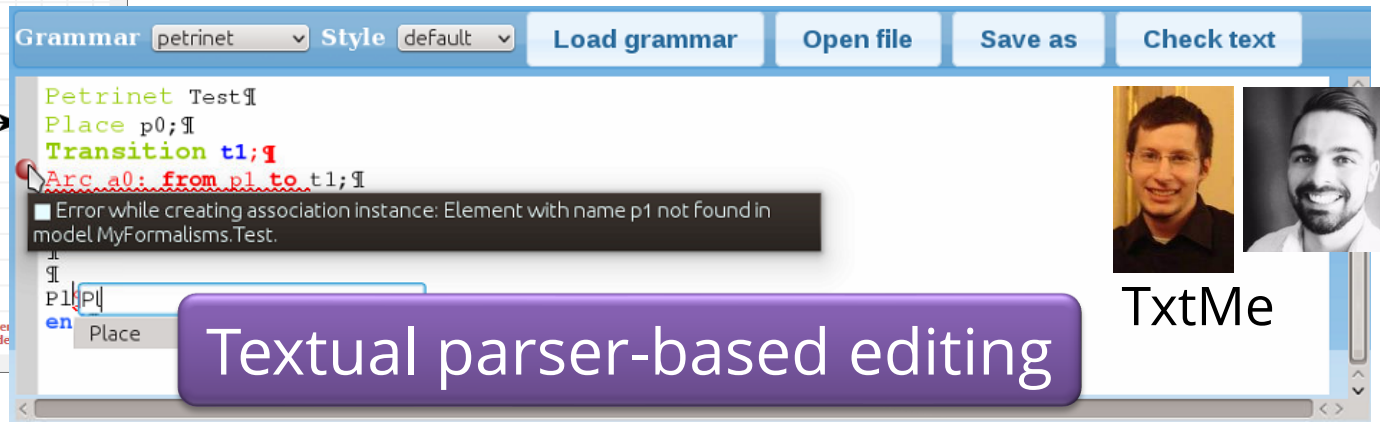
ATOMPM



Graphical syntax-directed editing

Projectional editing



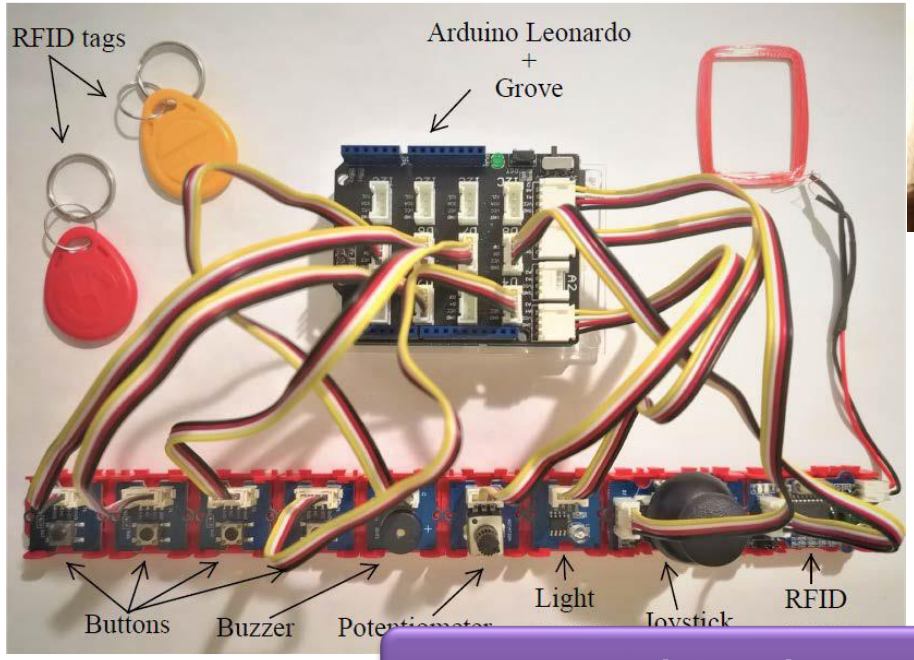


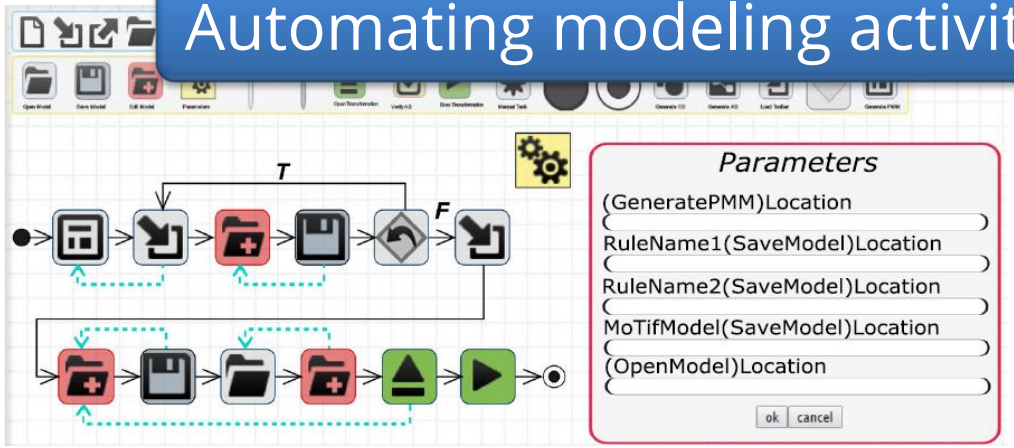
TxtMe

Textual parser-based editing

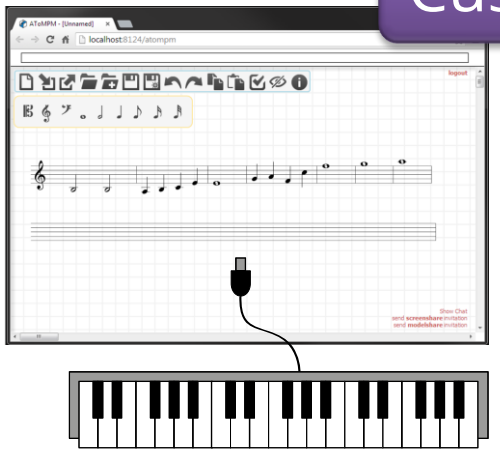
Domain-specific IDEs – Interaction

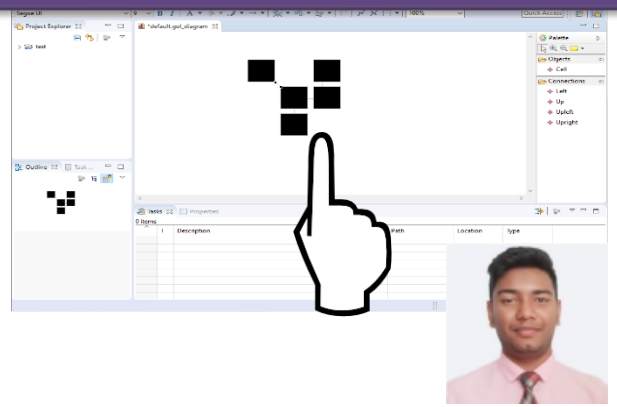
Automating modeling activities





Custom hardware for editors





```
InteractionRule StartPlay
  Condition {
    focus Interface playModelButton {}
  }
  --- press -->
  Effect {
    Interface light {value = "running"}
    Operation runGoL {}
    Interface playModelButton {value = "active"}
  }
```

```
InteractionRule TurnLightOff
  Condition {
    focus Interface
          lightButton {}
  }
  --- press -->
  Effect {
    Interface light {
          value = "off"
    }
  }
```

```
InteractionRule EndPlay
  Condition {
    Interface playModelButton {value = "active"}
  }
  --- _finish -->
  Effect {
    Interface light {value = "fi
    Interface playModelButton {
}
  }
```

Custom interactions

```
InteractionRule CreateCell
  Condition { focus Canvas {} } --- select --> Effect { Lang Cell {op = add} }
```

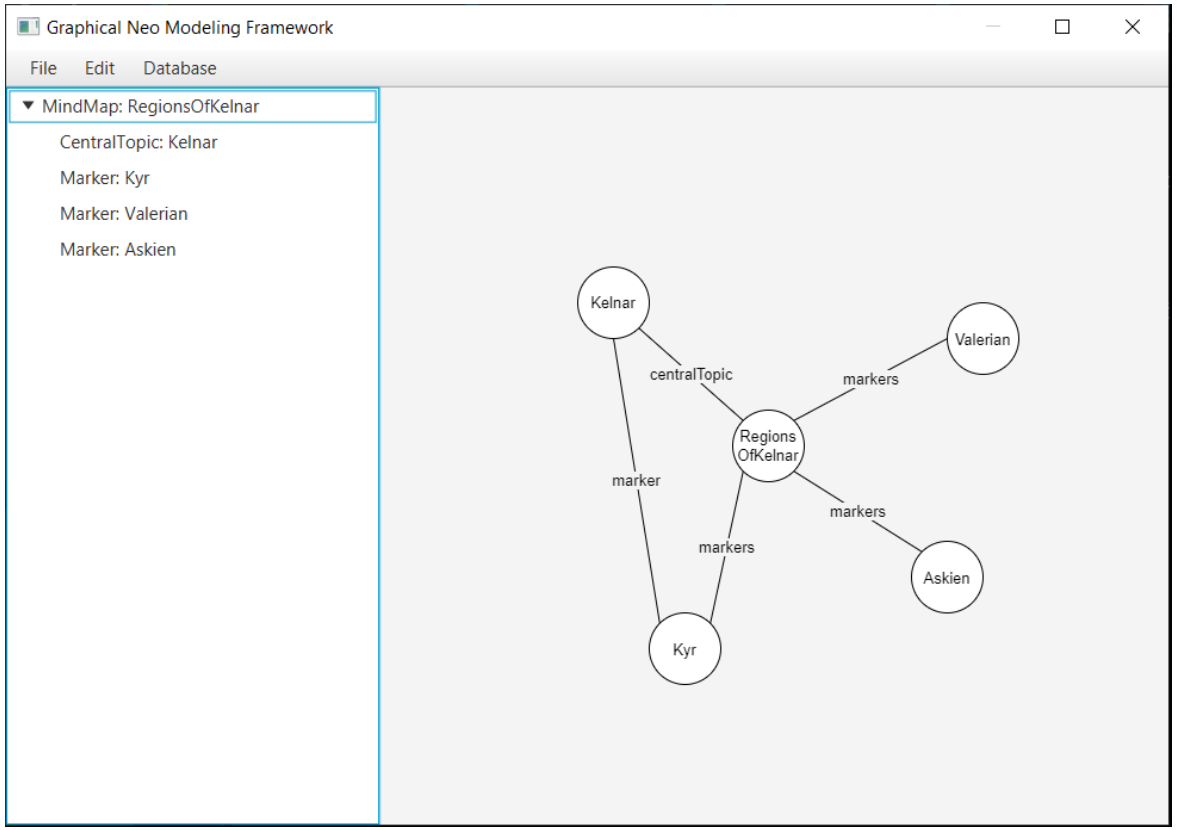
```
InteractionRule RemoveCell
  Condition { focus Lang Cell {} } --- select --> Effect { Lang Cell {op = rem} }
```

Domain-specific IDEs – Storage

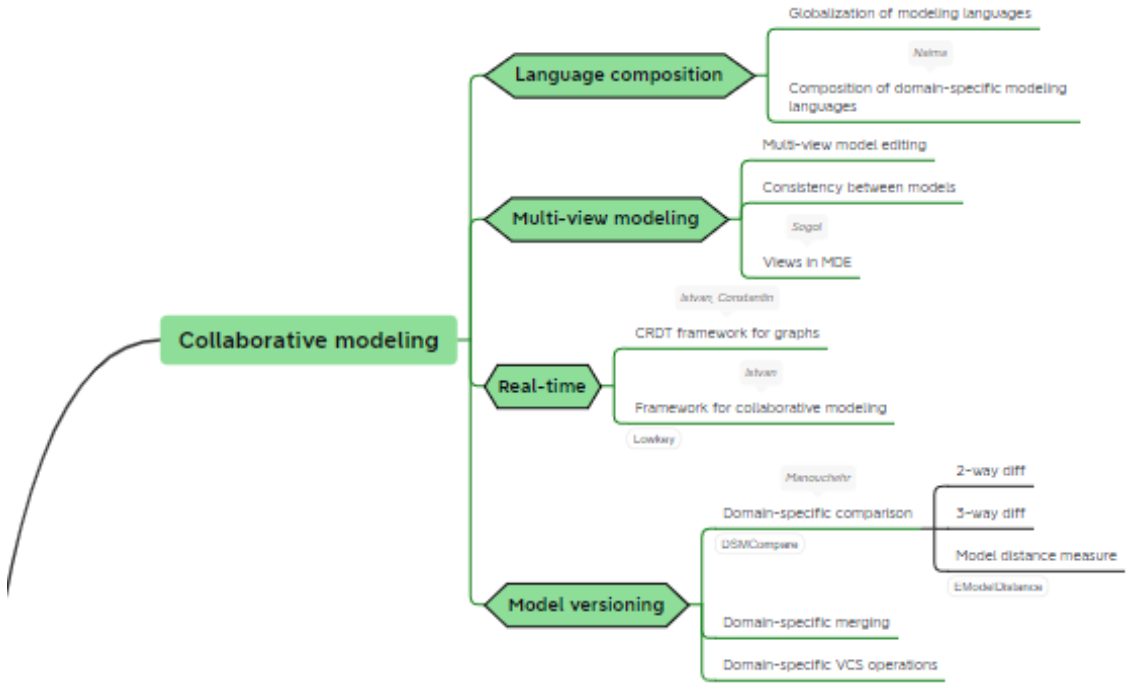
- Storing models in graph databases
- NeoModelingFramework

```
1  val manager = GrafModelManager(dbUri, username, password)
2  val graph = manager.createGraph()
3  graph.setName("G1")
4  val v0 = graph.addVertices(VertexType.CompositeVertex) as CompositeVertex
5  val v1 = v0.addSub_vertices(VertexType.Vertex)
6  val v2 = v0.addSub_vertices(VertexType.Vertex)
7  v0.setDefault_vertex(v1)
8  manager.saveChanges()
9  manager.close()
```

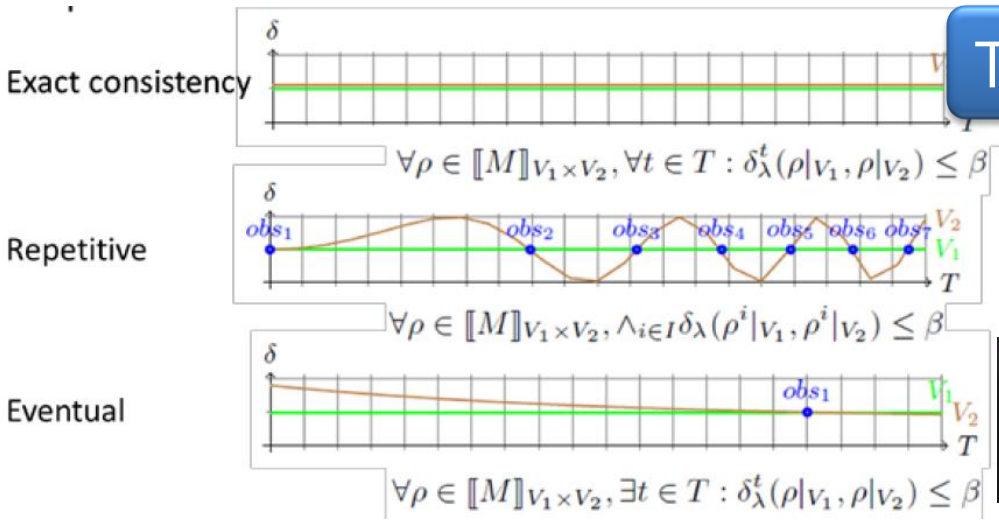
Domain-specific API



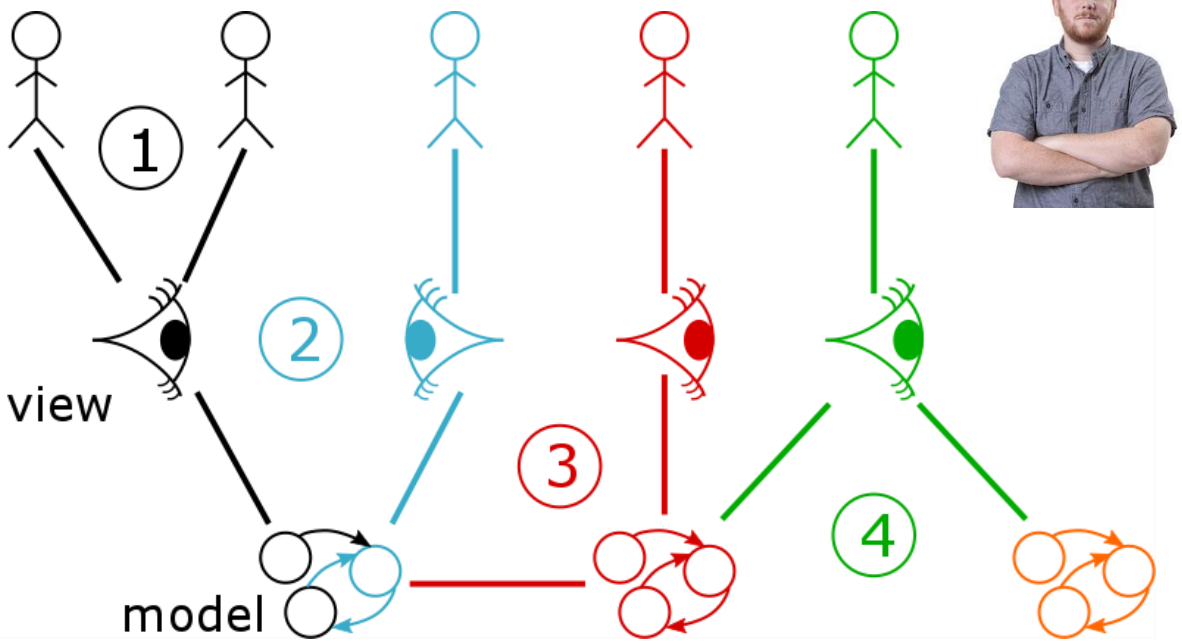
Outline

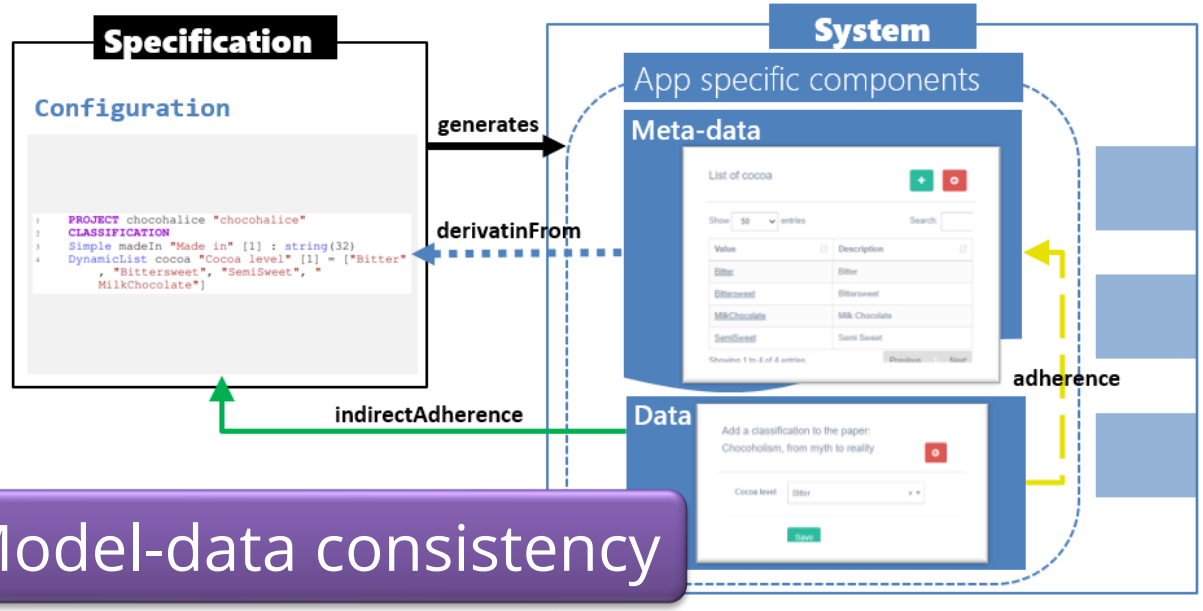

Collaborative modeling
Language composition
Globalization of modeling languages
Composition of domain-specific modeling languages
Multi-view modeling
Multi-view model editing
Consistency between models
Views in MDE
Real-time
CRDT framework for graphs
Framework for collaborative modeling
Model versioning
Domain-specific comparison
2-way diff
3-way diff
Model distance measure
Domain-specific merging
Domain-specific VCS operations

Collaborative modeling – Multi-view consistency



Tolerating inconsistency

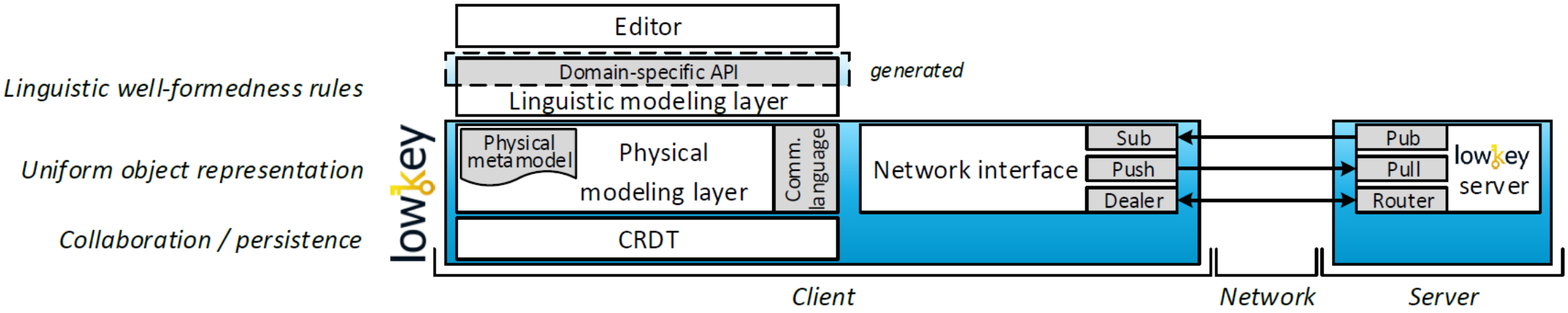




Model-data consistency

Multi-view modeling

Collaborative modeling – Real-time collaboration

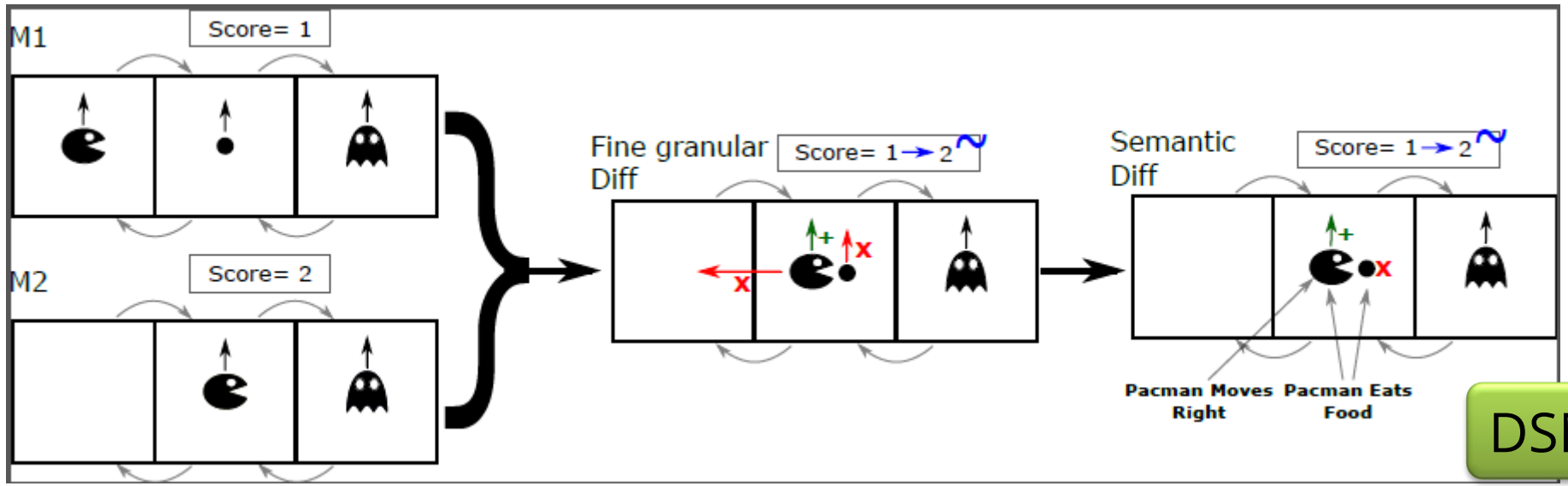


CollabServer

CRDT

lowkey

Collaborative modeling – Model versioning



DSMCompare

Move $$\Delta_M(G_1, G_2) = \sum_{m_1 \in Mov_1, m_2 \in Mov_2} \delta_M(m_1, p(m_2)), \text{ where } l(m_1) = l(m_2)$$

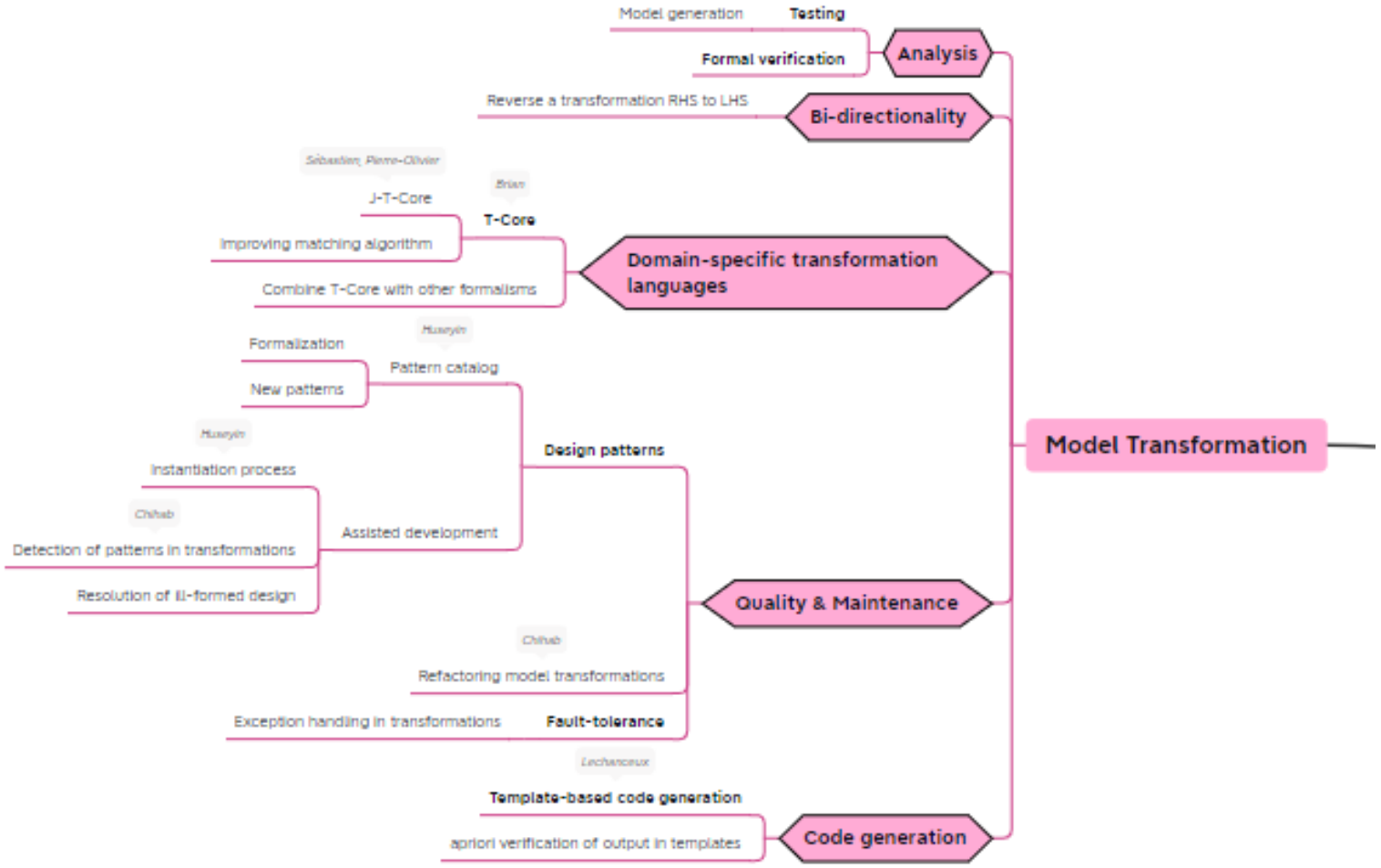
Element $$\Delta_E(G_1, G_2) = \frac{|\{v_1 \in V_1 | \nexists v_2 \in V_2, l(v_2) = l(v_1)\}| + |\{v_2 \in V_2 | \nexists v_1 \in V_1, l(v_1) = l(v_2)\}|}{|V_1| + |V_2|}$$

Value $$\delta_V(v, x) = \begin{cases} |a(v_1, x)| & \text{if } a(v, x) = 0 \\ |a(v, x) - a(v_1, x)|/a(v, x) & \text{otherwise} \end{cases}$$

TU WIEN

Domain-specific distance

Outline


Model Transformation
Analysis
Model generation
Testing
Formal verification
Bi-directionality
Reverse a transformation RHS to LHS
Domain-specific transformation languages
Sébastien, Pierre-Olivier
J-T-Core
Brian
T-Core
Improving matching algorithm
Combine T-Core with other formalisms
Quality & Maintenance
Design patterns
Huseyin
Formalization
Pattern catalog
New patterns
Huseyin
Instantiation process
Chihab
Assisted development
Detection of patterns in transformations
Resolution of ill-formed design
Chihab
Refactoring model transformations
Exception handling in transformations
Fault-tolerance
Code generation
Lechanceux
Template-based code generation
apriori verification of output in templates

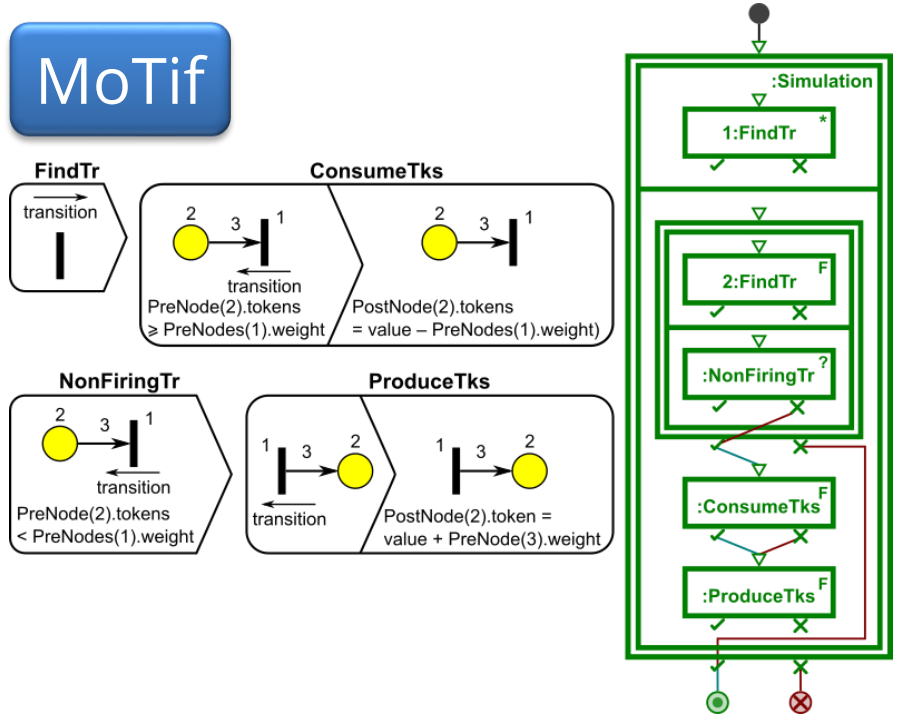
Model transformation – Domain-specific transformation languages

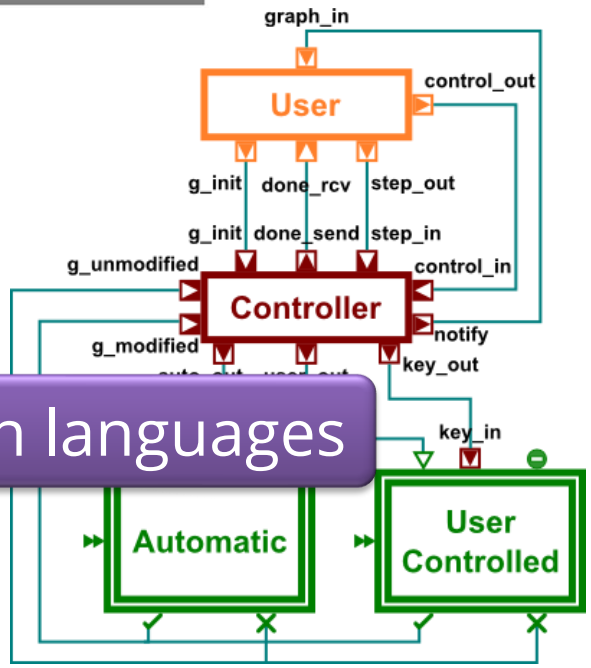
T-Core

```
class ARule(Composer):
  def __init__(self, LHS, RHS):
    super(ARule, self).__init__()
    self.M = Matcher(condition=LHS, max=1)
    self.I = Iterator(max_iterations=1)
    self.W = Rewriter(condition=RHS)
  def packet_in(self, packet):
    self.is_success = False
    packet = self.M.packet_in(packet)
    if not self.M.is_success: return packet
    packet = self.I.packet_in(packet)
    if not self.I.is_success: return packet
    packet = self.W.packet_in(packet)
    if not self.W.is_success: return packet
    self.is_success = True
    return packet
```

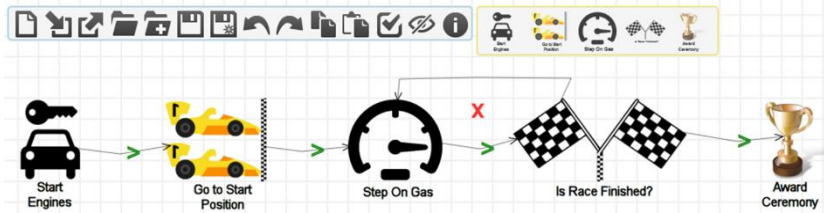
McGill

MoTif





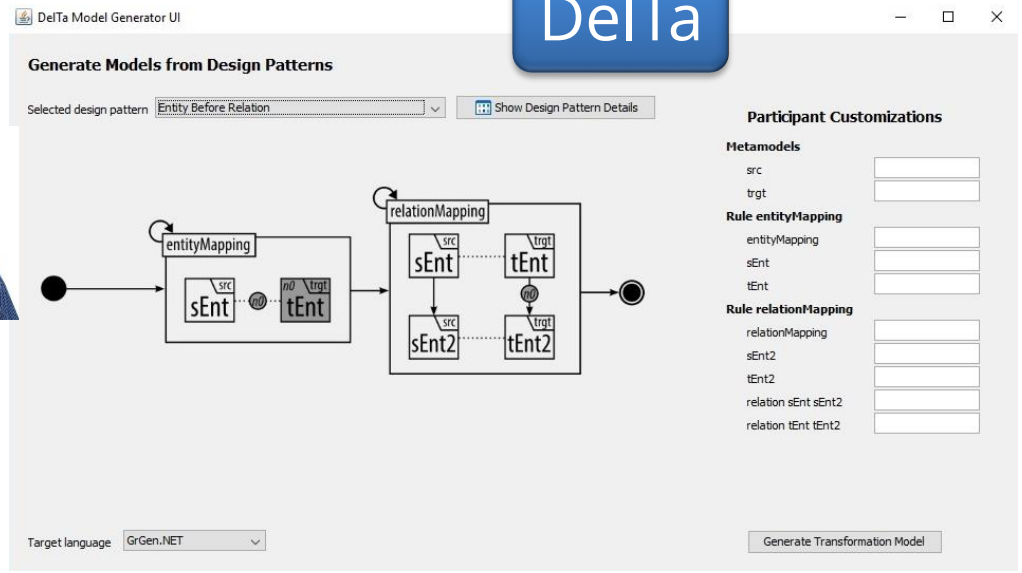
Custom transformation languages



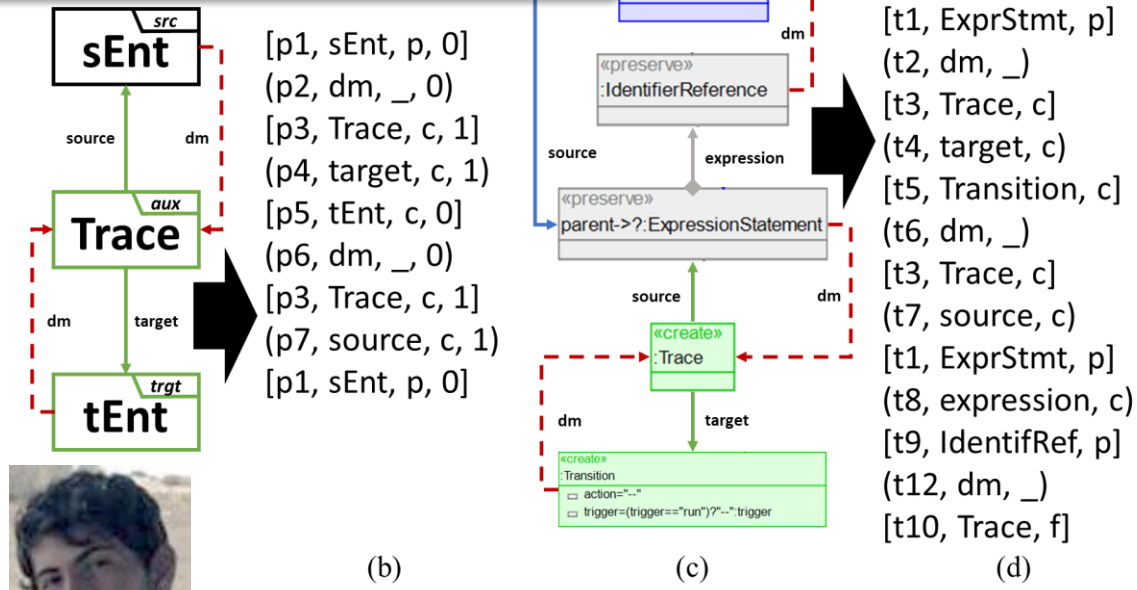
Higher-order transformations

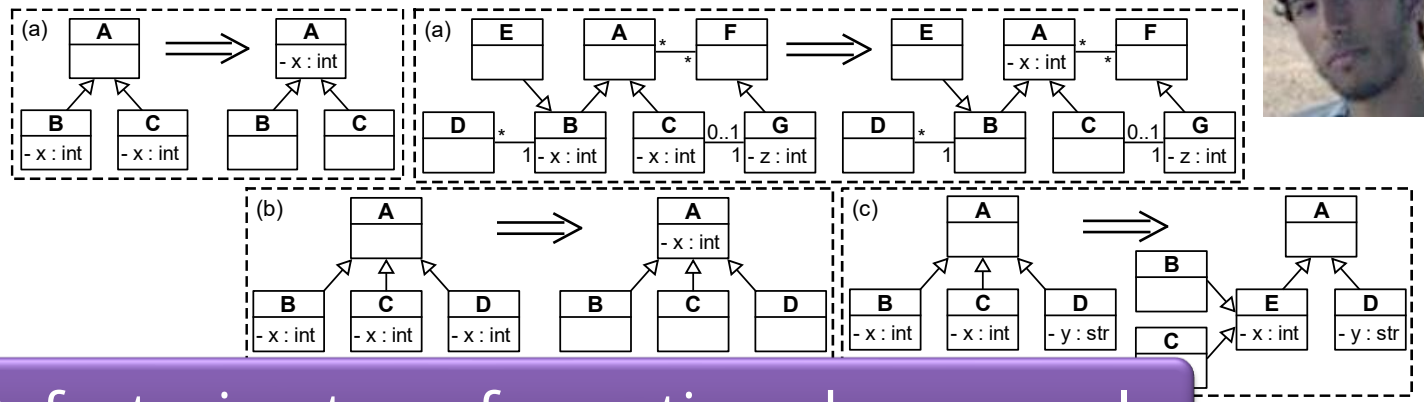
Model transformation – Design patterns

DelTa



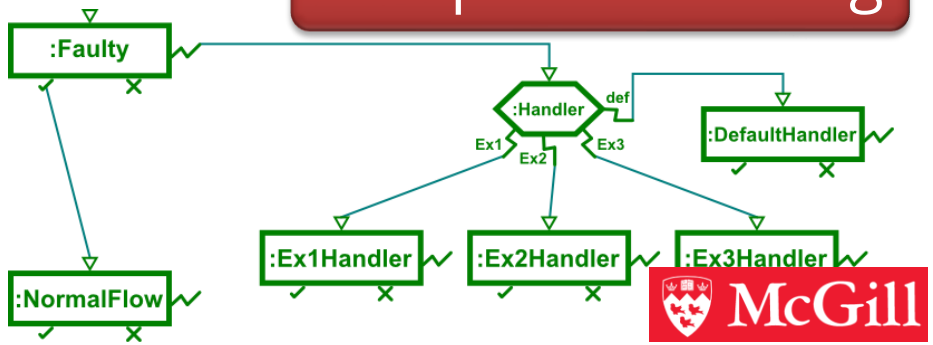
Detection of variants





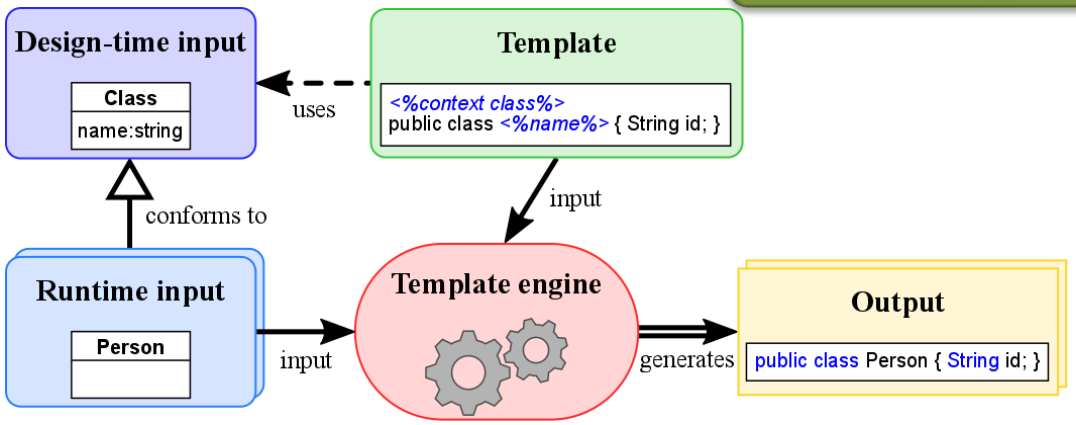
Refactoring transformations by-example

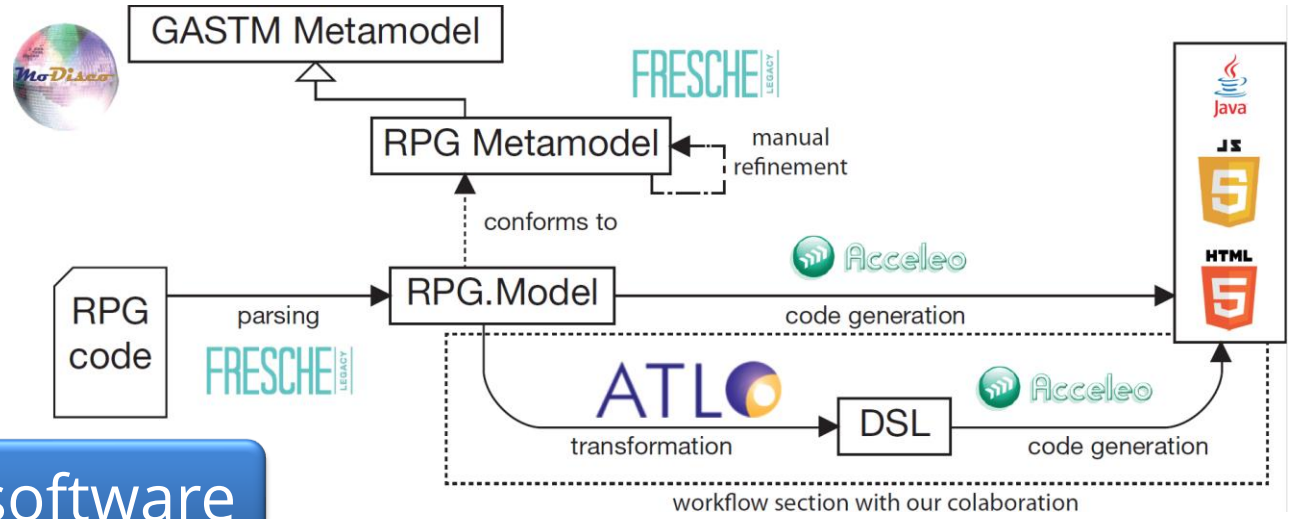
Exception handling



Model transformation – Code generation

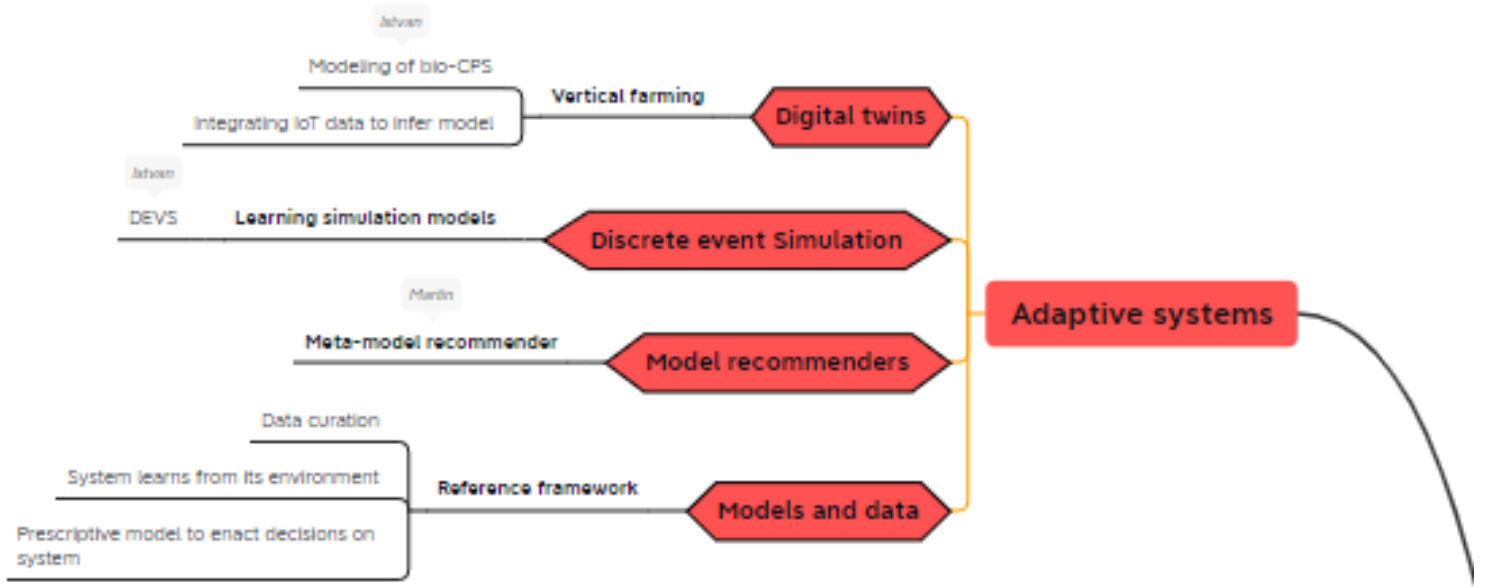
Template-based code generation





Modernization of legacy software

Outline

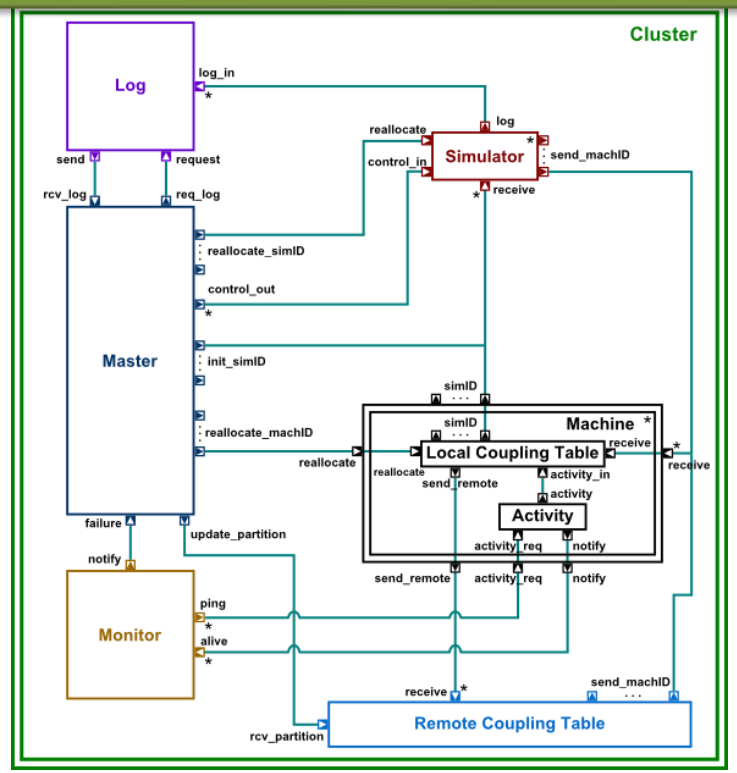

Adaptive systems
Digital twins
Vertical farming
Modeling of bio-CPS
Integrating IoT data to infer model
Discrete event Simulation
Learning simulation models
DEVS
Model recommenders
Meta-model recommender
Models and data
Reference framework
Data curation
System learns from its environment
Prescriptive model to enact decisions on system

Adaptive systems – Simulation

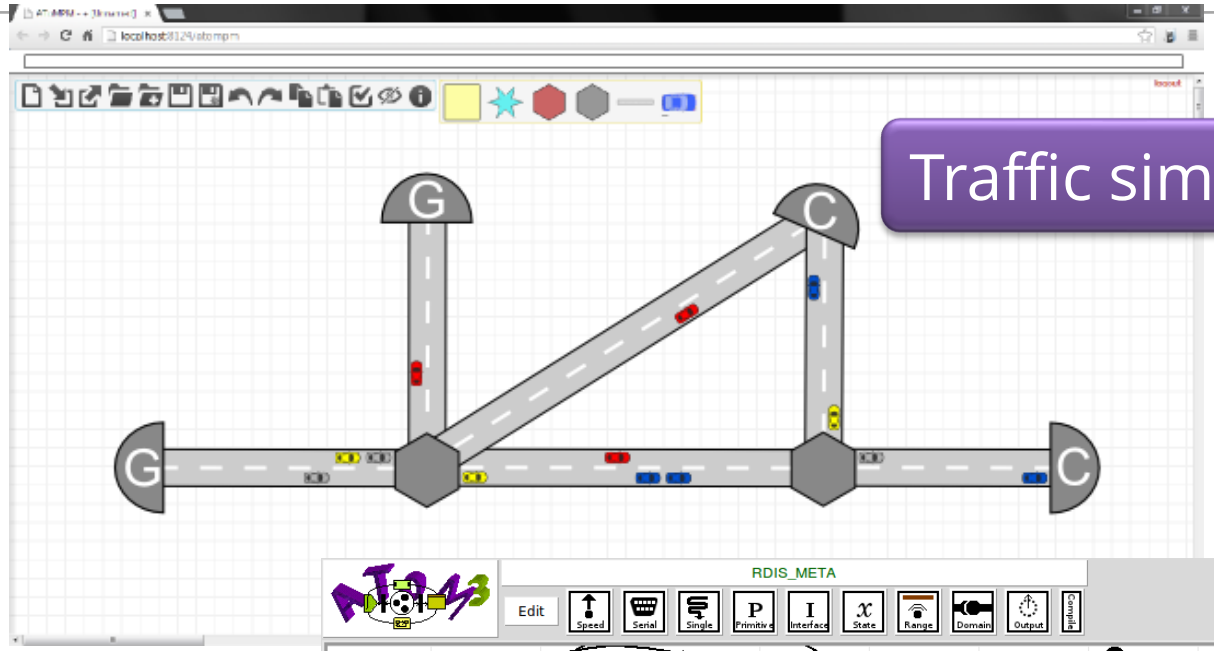
THE UNIVERSITY OF ALABAMA

McGill

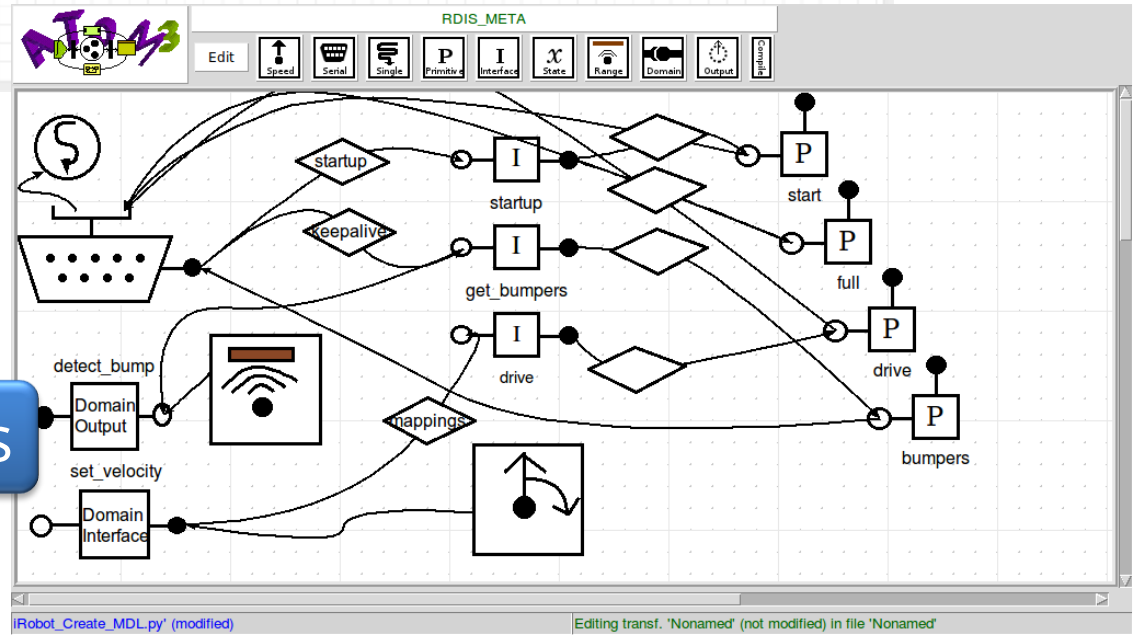
Distributed DEVS simulators



Traffic simulation



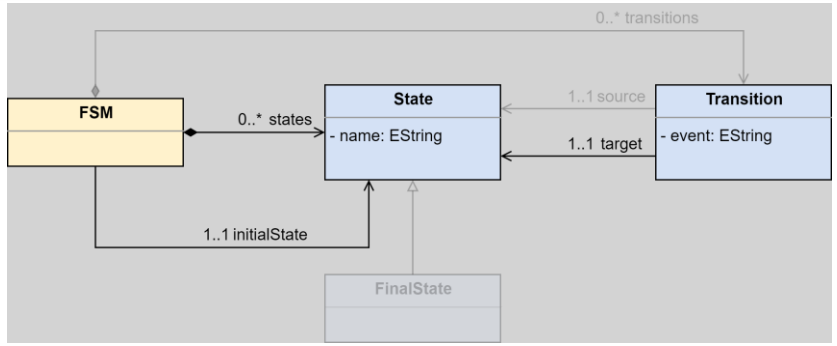
Robotics

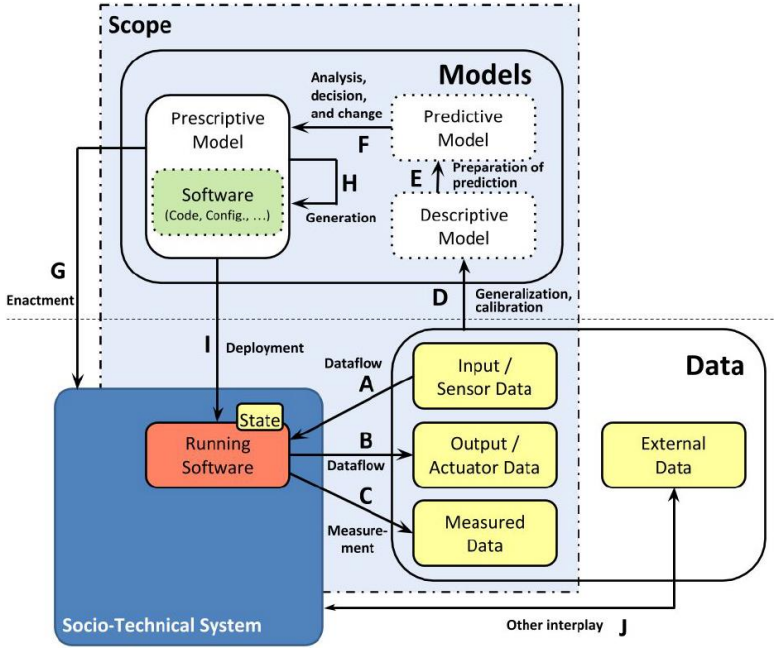


Adaptive systems – Data and machine learning

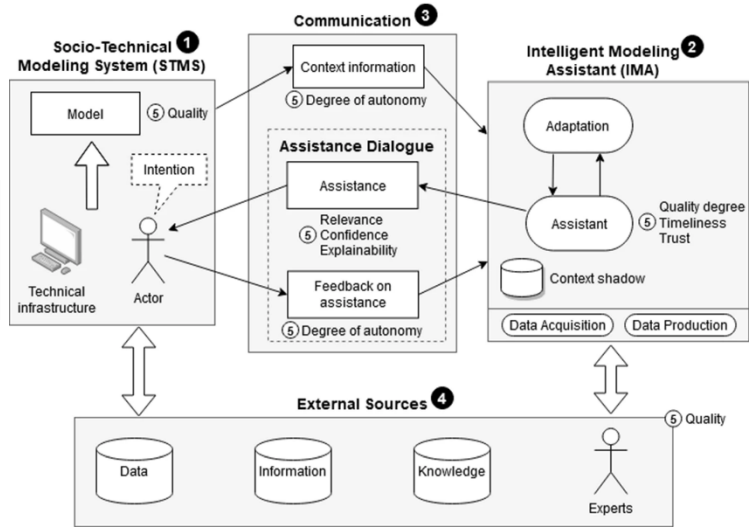
Meta-model recommenders


B. Combemale, J. Kienzle, G. Mussbacher, H. Ali, D. Amyot, M. Bagherzadeh, E. Batot, N. Bencomo, B. Benni, J. Bruel, J. Cabot, B. Cheng, P. Collet, G. Engels, R. Heinrich, J. Jezequel, A. Koziolek, S. Mosser, R. Reussner, H. Sahraoui, R. Saini, J. Sallou, S. Stinckwich, E. Syriani, and M. Wimmer. A Hitchhiker's Guide to Model-Driven Engineering for Data-Centric Systems. *IEEE Software*: 38(4), pp. 71–84 (2020).





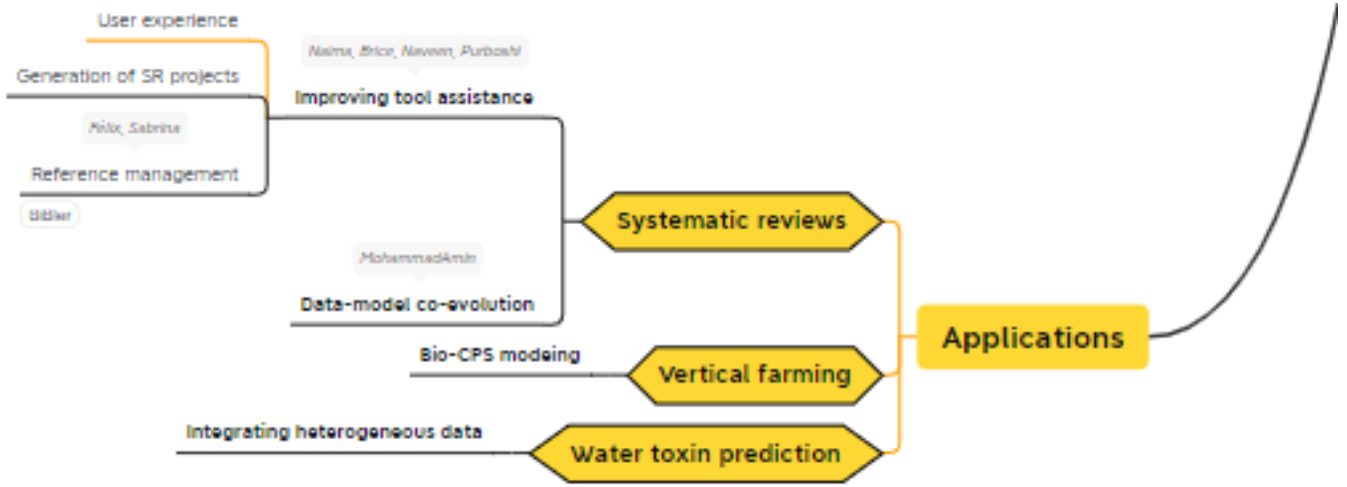
G. Mussbacher, B. Combemale, J. Kienzle, S. Abrahão, H. Ali, N. Bencomo, M. Búr, L. Burgueño, G. Engels, P. Jeanjean, J. Jézéquel, T. Kühn, S. Mosser, H. Sahraoui, E. Syriani, D. Varró, and M. Weyssow. Opportunities in Intelligent Modeling Assistance. *Software and Systems Modeling*: 19, pp. 1045–1053. (2020).



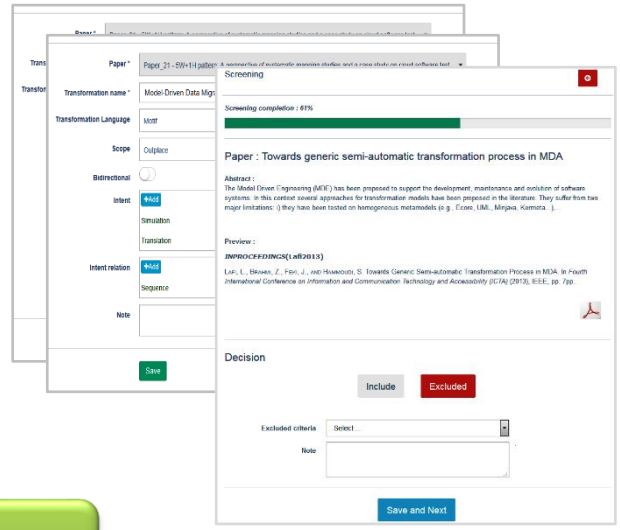
Models and data reference framework

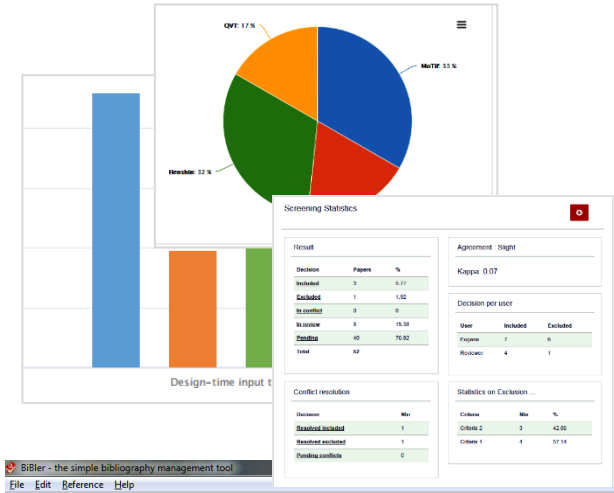
Intelligent modeling assistance

Outline

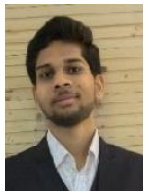

User experience
Generation of SR projects
Naima, Brice, Naveen, Purbosh
Improving tool assistance
Felix, Sabrina
Reference management
Didier
MohammadAmin
Data-model co-evolution
Systematic reviews
Bio-CPS modeling
Vertical farming
Integrating heterogeneous data
Water toxin prediction
Applications

Systematic reviews

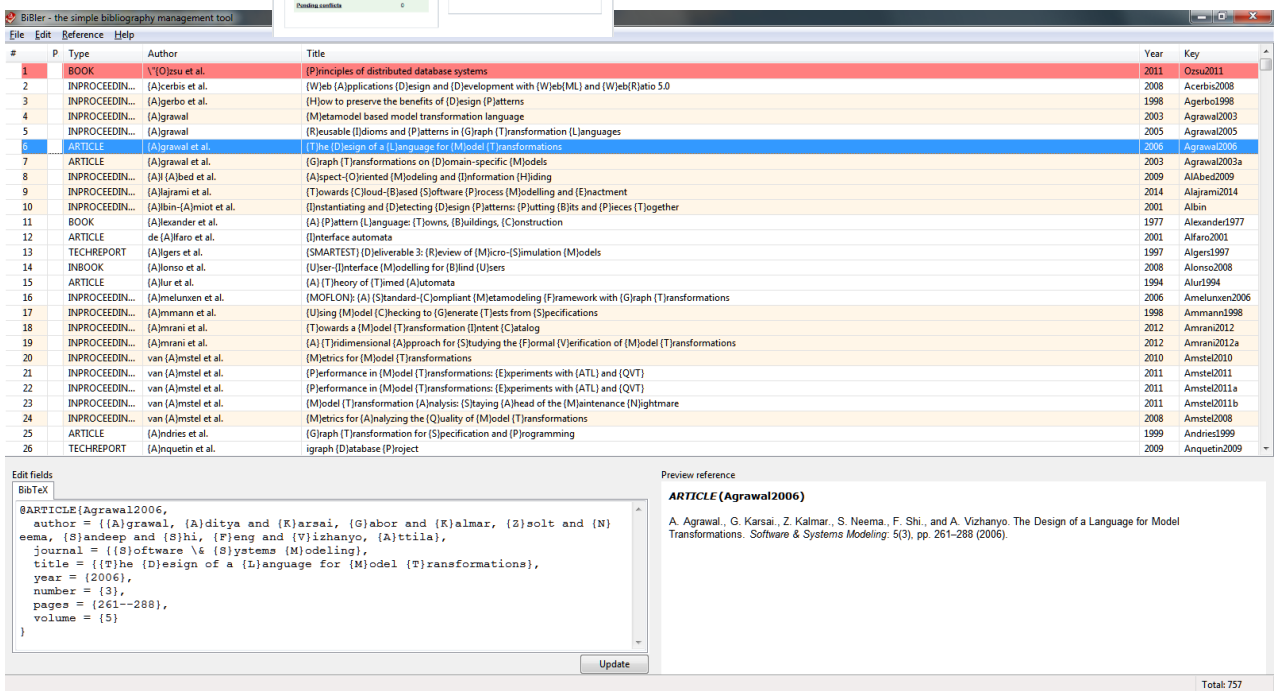




ReLiS



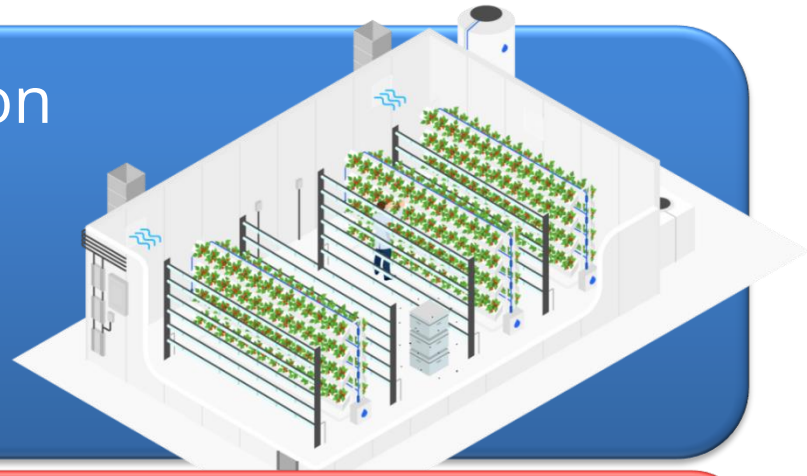
BiBler



Upcoming projects

Bio-Cyber-Physical System modeling and simulation

Vertical farming

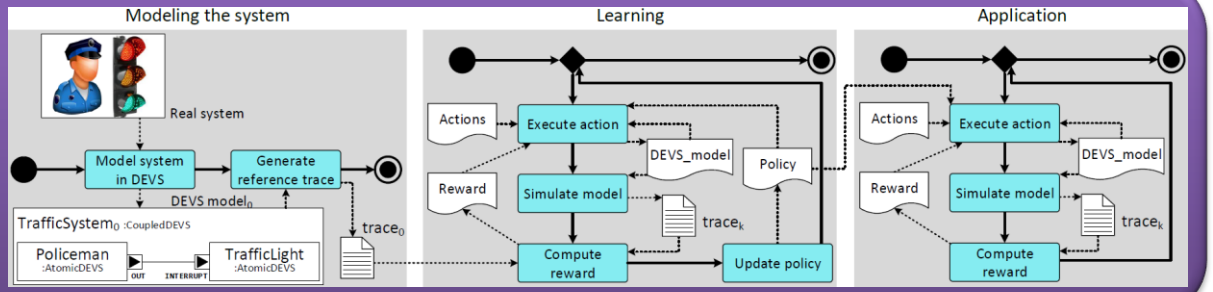


Integration of heterogeneous data for AI model validation

Water toxin prediction

Learning DEVS simulation models

Digital twins



Overview of our research in software modeling & simulation