Exploiting Traditional Versioning Operators for Managing Variability in Model-based Software

Experiences and Future Perspectives

Timo Kehrer[1], Christopher Pietsch[2], Dennis Reuling[2]

[1]Department of Computer Science
Humboldt-University of Berlin

[2]Software Engineering Group
University of Siegen, Germany

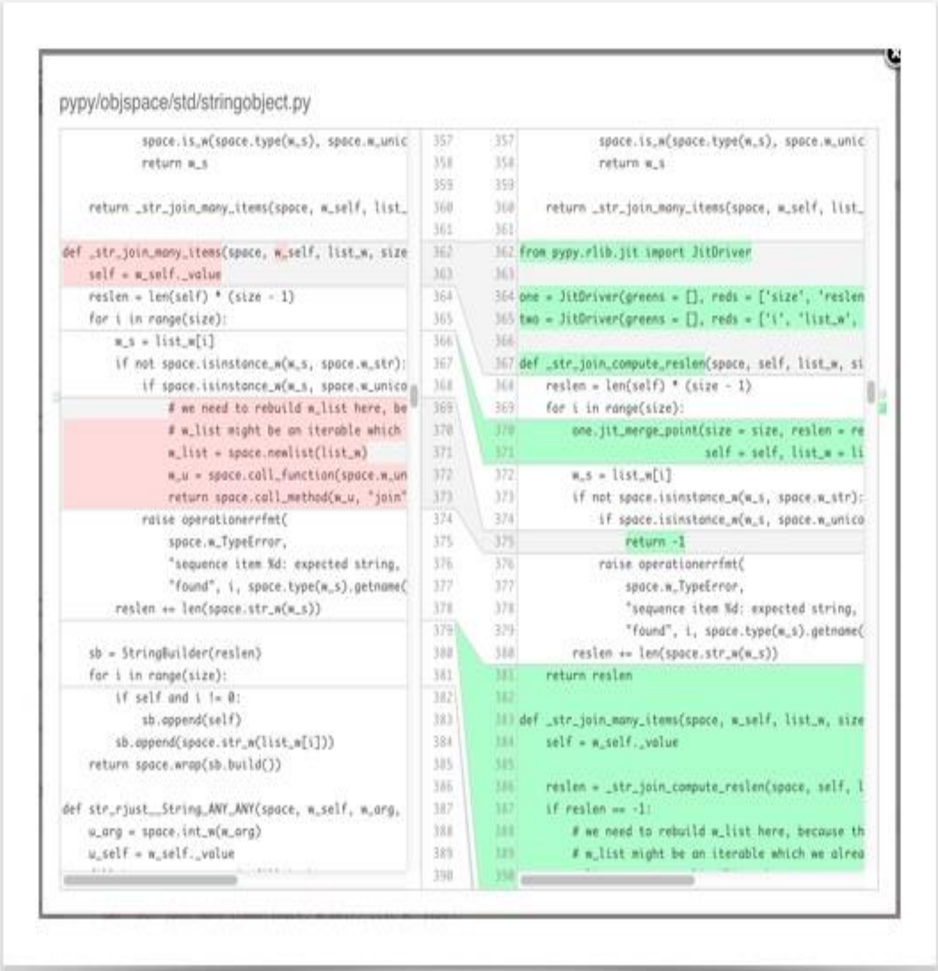
FOSD 2017 Meeting

14-17 March, 2017, Technische Universität Darmstadt

From Traditional Text-based Versioning to Model Versioning

Sample extraction from the Subversion command list:

```
[kehrer@tkmobil1 ~]$ svn help
   checkout (co)
   commit (ci)
   diff (di)
   info
   lock
   log
   merge
   patch
   revert
   status (stat, st)
   switch (sw)
   unlock
   update (up)
```



pypy/objspace/std/stringobject.py

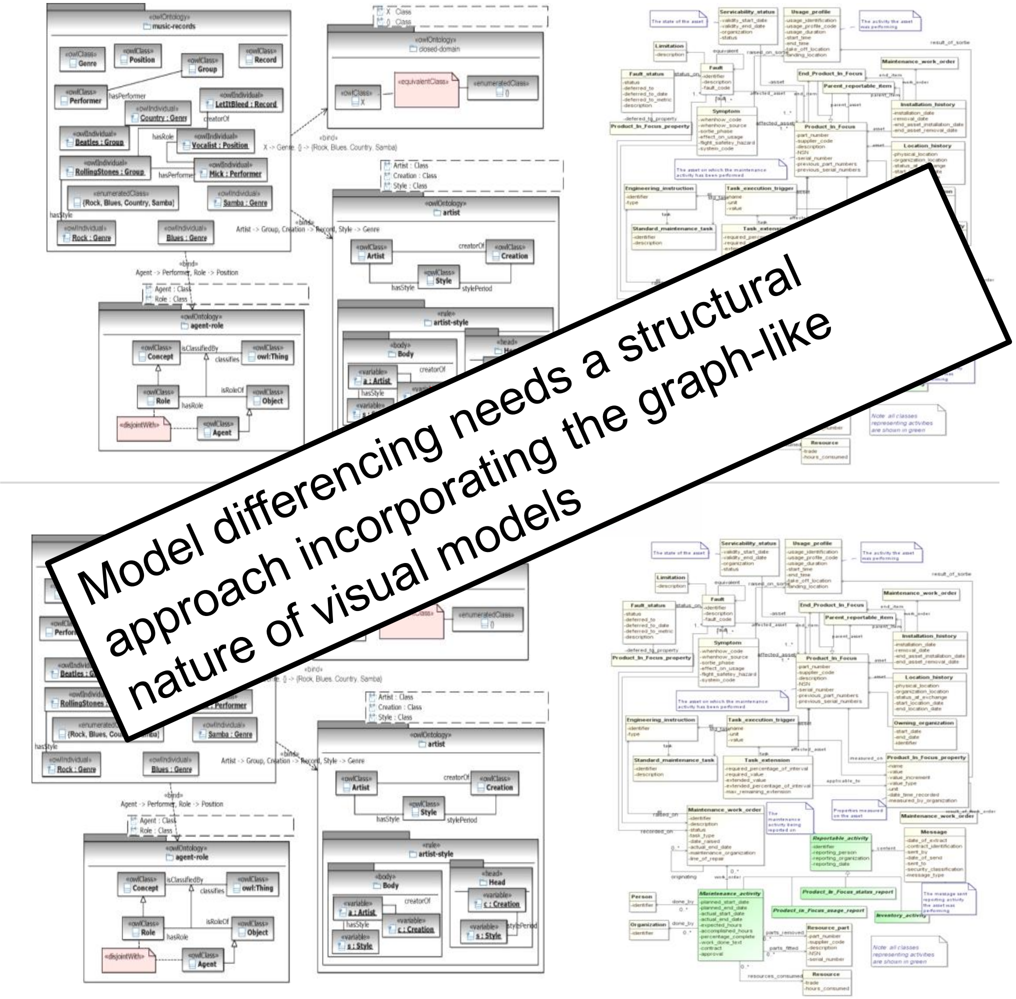
```
            space.is_w(space.type(w_s), space.w_unic
            return w_s

    return _str_join_many_items(space, w_self, list_

def _str_join_many_items(space, w_self, list_w, size
    self = w_self._value
    reslen = len(self) * (size - 1)
    for i in range(size):
        w_s = list_w[i]
        if not space.isinstance_w(w_s, space.w_str):
            if space.isinstance_w(w_s, space.w_unico
                # we need to rebuild w_list here, be
                # w_list might be an iterable which
                w_list = space.newlist(list_w)
                w_u = space.call_function(space.w_un
                return space.call_method(w_u, "join"
            raise operationerrfmt(
                space.w_TypeError,
                "sequence item %d: expected string,
                "found", i, space.type(w_s).getname(
        reslen += len(space.str_w(w_s))

    sb = StringBuilder(reslen)
    for i in range(size):
        if self and i != 0:
            sb.append(self)
        sb.append(space.str_w(list_w[i]))
    return space.wrap(sb.build())

def str_rjust__String_ANY_ANY(space, w_self, w_arg,
    u_arg = space.int_w(w_arg)
    u_self = w_self._value
```

```
            space.is_w(space.type(w_s), space.w_unic
            return w_s

    return _str_join_many_items(space, w_self, list_

from pypy.rlib.jit import JitDriver

one = JitDriver(greens = [], reds = ['size', 'reslen
two = JitDriver(greens = [], reds = ['i', 'list_w',

def _str_join_compute_reslen(space, self, list_w, si
    reslen = len(self) * (size - 1)
    for i in range(size):
        one.jit_merge_point(size = size, reslen = re
                            self = self, list_w = li
        w_s = list_w[i]
        if not space.isinstance_w(w_s, space.w_str):
            if space.isinstance_w(w_s, space.w_unico
                return -1
            raise operationerrfmt(
                space.w_TypeError,
                "sequence item %d: expected string,
                "found", i, space.type(w_s).getname(
        reslen += len(space.str_w(w_s))
    return reslen

def _str_join_many_items(space, w_self, list_w, size
    self = w_self._value

    reslen = _str_join_compute_reslen(space, self, l
    if reslen == -1:
        # we need to rebuild w_list here, because th
        # w_list might be an iterable which we alrea
```

From Traditional Text-based Versioning to Model Versioning

Sample extraction from the Subversion command list:

```
[kehrer@tkmobil1 ~]$ svn help
   checkout (co)
   commit (ci)
   diff (di)
   info
   lock
   log
   merge
   patch
   revert
   status (stat, st)
   switch (sw)
   unlock
   update (up)
```



Model differencing needs a structural approach incorporating the graph-like nature of visual models

Research on Model Versioning in Several Research Projects

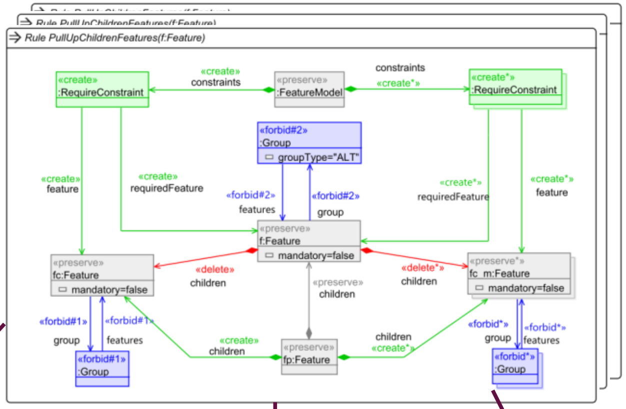
sidiff see the difference [2005 - 2011]

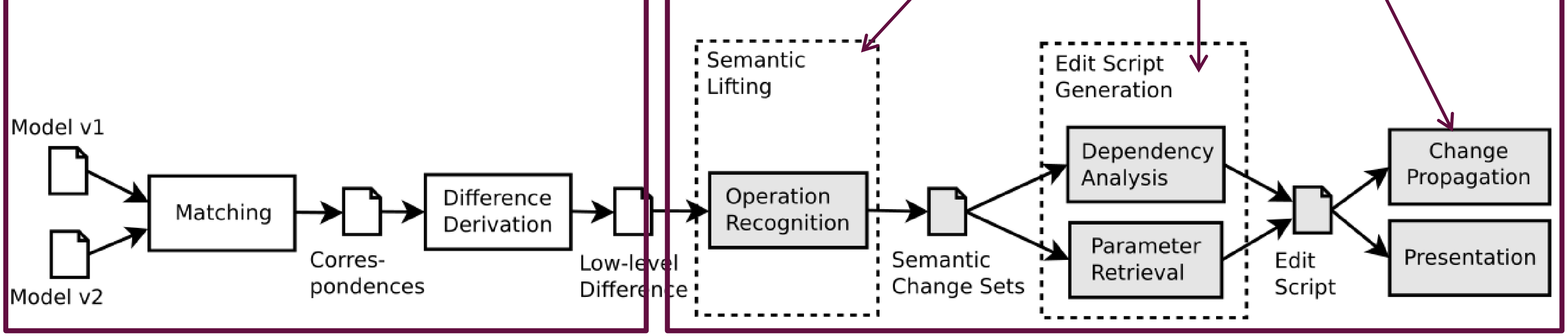
Results:

- A set of configurable incremental matchers
- A DSL to control their application

MOCA DFG Design For FUTURE [2012 - 2015]

Extended differencing pipeline configured by formal specifications of editing operations





Research on Model Versioning in Several Research Projects

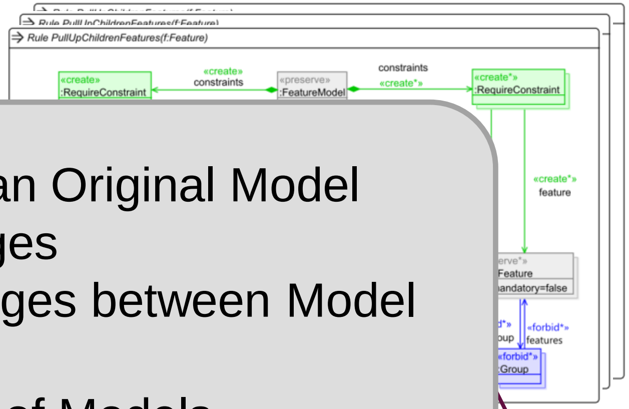
sidiff see the difference [2005 - 2011]

Results:

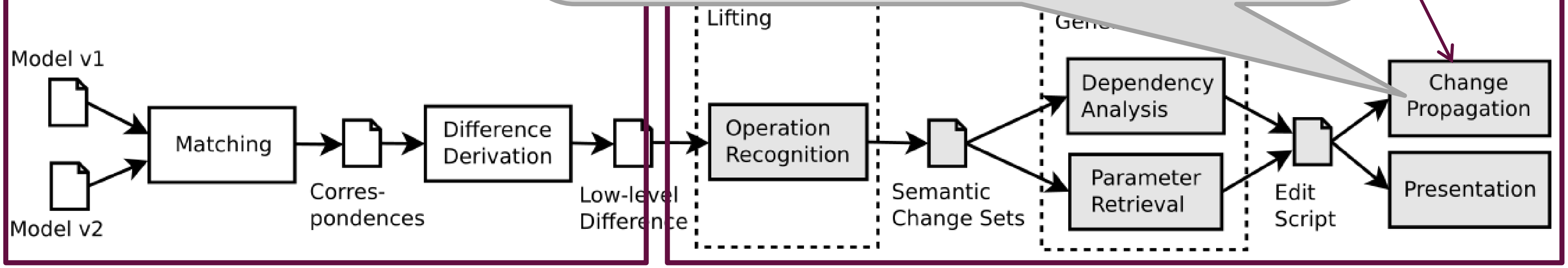
- A set of configurable incremental matchers
- A DSL to control their application

MOCA DFG Design For FUTURE [2012 - 2015]

Extended differencing pipeline configured by formal specifications of editing operations



- Updating a Copy of an Original Model
- Cherrypicking Changes
- Propagation of Changes between Model Variants
- Workspace Updates of Models



Unlocks Potential for Exploiting Graph Transformation Concepts

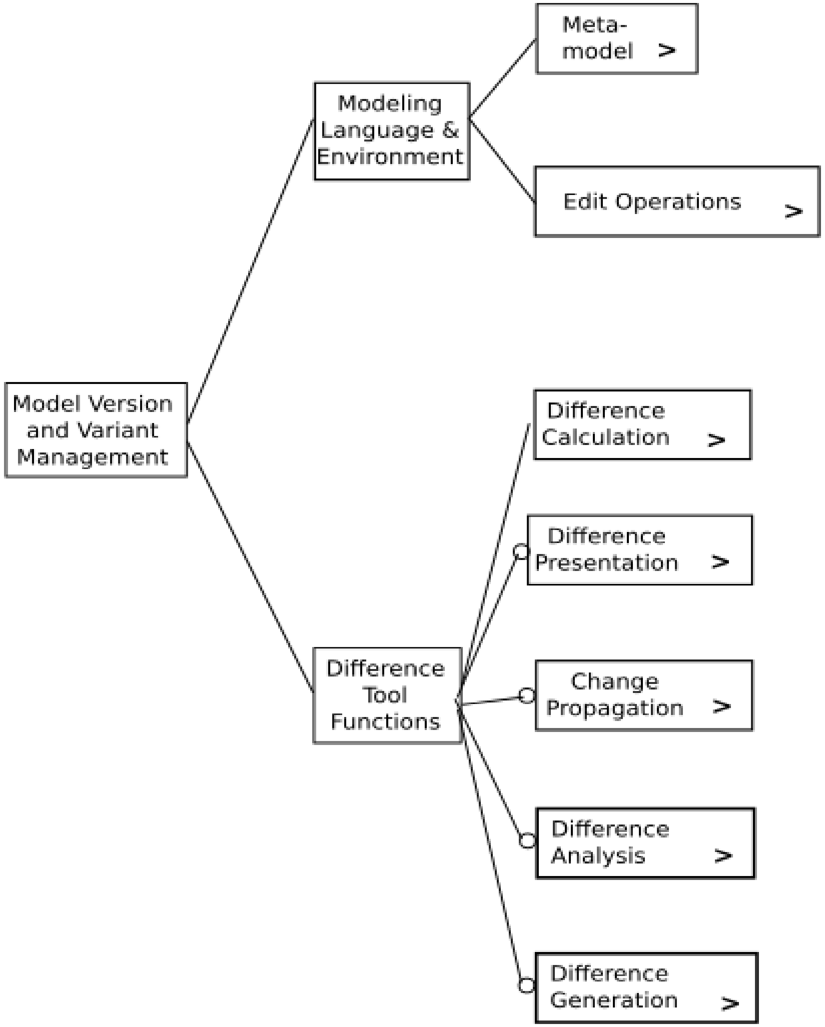
Formal semantics and declarative nature of transformation rules

$$\begin{tikzcd}
L \arrow[d, "m"'] & K \arrow[l] \arrow[r] \arrow[d] & R \arrow[d] \\
G \arrow[ur, phantom, "(1)"] & D \arrow[l] \arrow[r] \arrow[ur, phantom, "(2)"] & H
\end{tikzcd}$$

Critical pair analysis, potential conflicts and dependencies

$$\begin{tikzcd}
 & G \arrow[dl, "p_1(o_1)"'] \arrow[dr, "p_2(o_2)"] & \\
H_1 \arrow[dr, "p_2(o_2)"'] & & H_2 \arrow[dl, "p_1(o_1)"] \\
 & X &
\end{tikzcd}$$

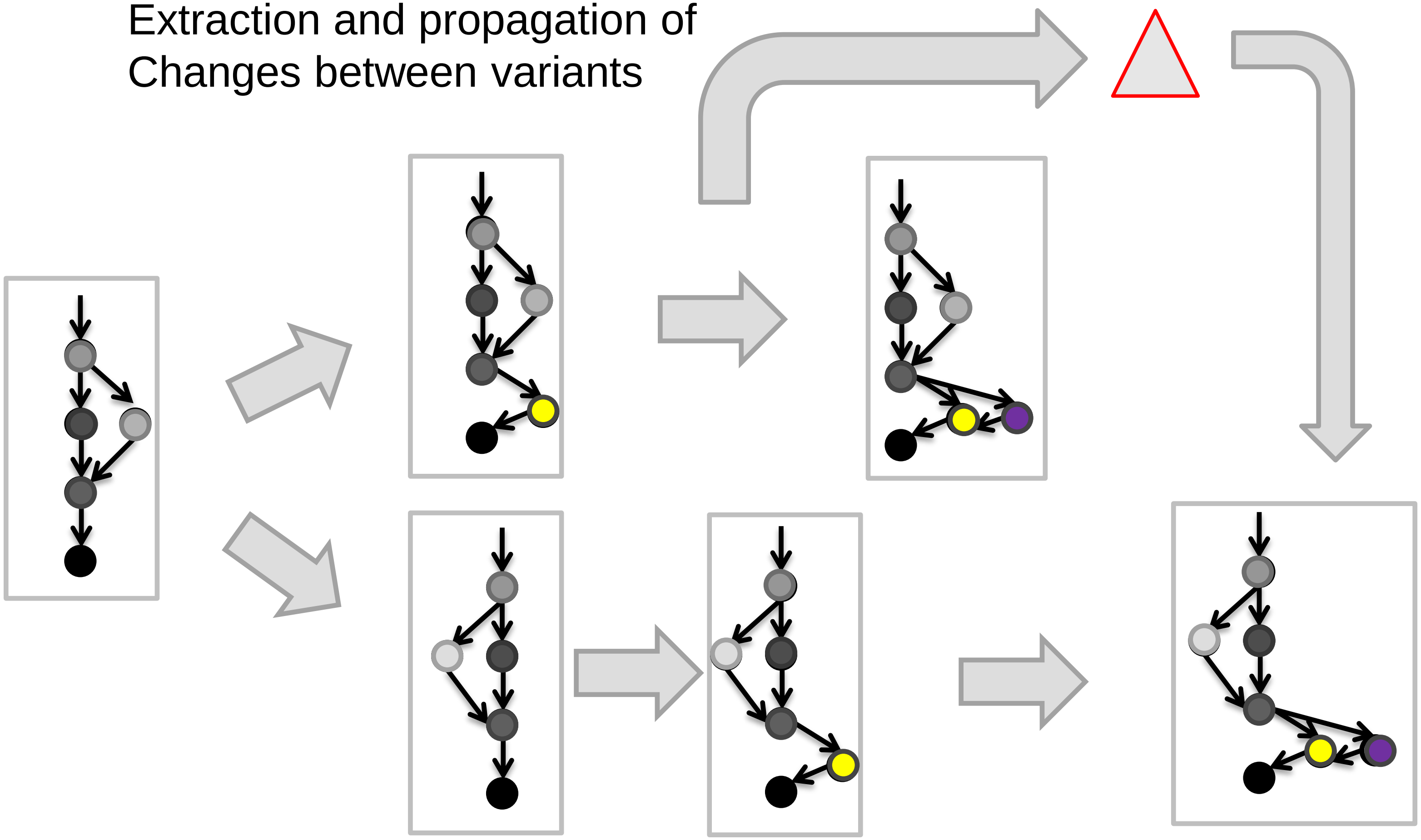
A Family of Flexible and Adaptable Versioning Operators


Model Version and Variant Management
Modeling Language & Environment
Meta-model >
Edit Operations >
Difference Tool Functions
Difference Calculation >
Difference Presentation >
Change Propagation >
Difference Analysis >
Difference Generation >

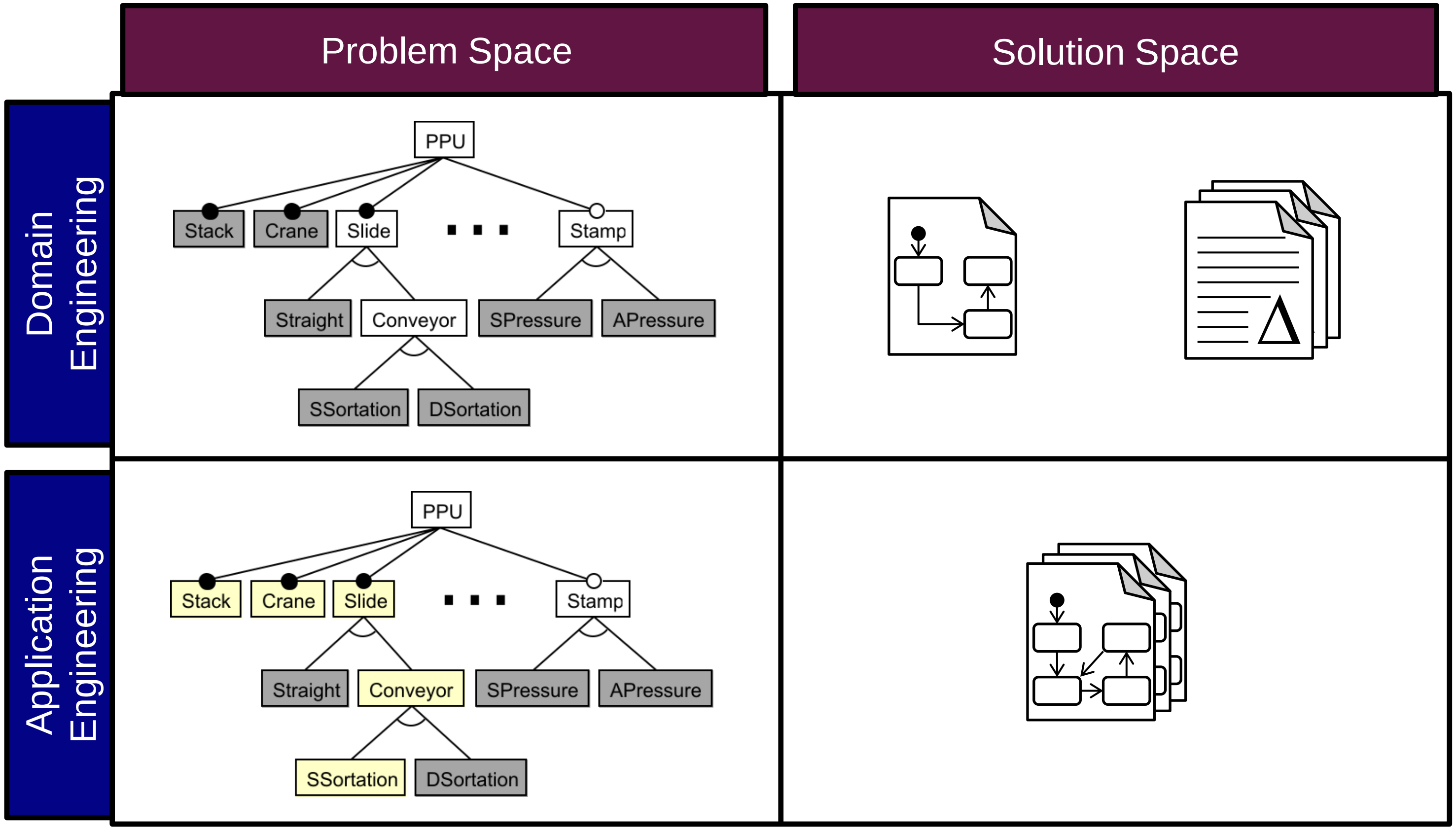
Adaptation to modeling language and environment

Adaptation to use case and application scenario

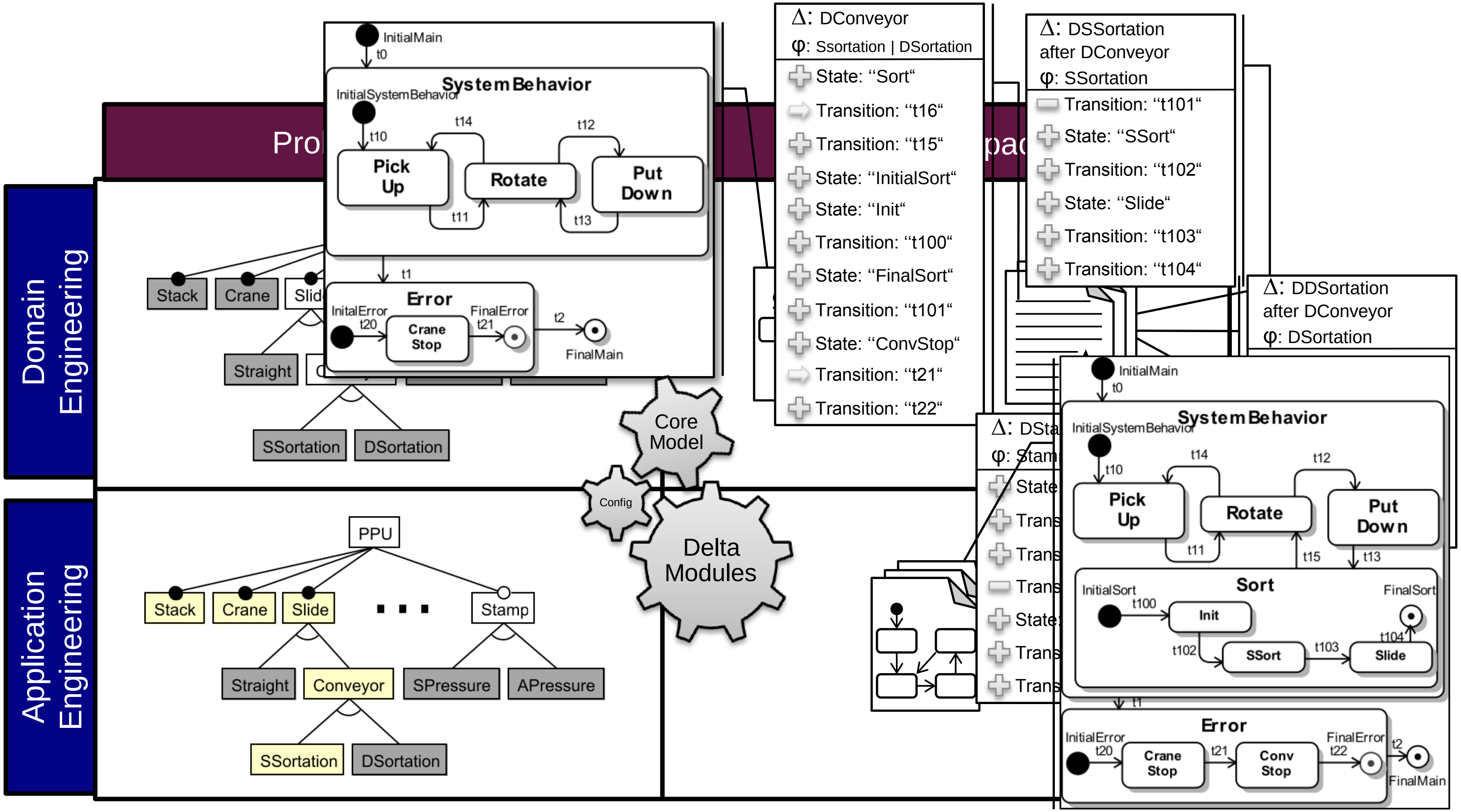
Instantiation for Ad-hoc Management of Variants Evolving in Parallel



Instantiation in the Context of Delta-oriented MBSPLE



Instantiation in the Context of Delta-oriented MBSPLE



Domain Engineering

Application Engineering

Prob

pac

Stack Crane Slid

Straight

SSortation DSortation

InitialMain
t0
System Behavior
InitialSystemBehavior
t10 t14 t12
Pick Up
Rotate
Put Down
t11 t13
t1
Error
InitalError
t20
Crane Stop
FinalError
t21
t2
FinalMain

Core Model

Config

Delta Modules

PPU

Stack Crane Slide . . . Stamp

Straight Conveyor SPressure APressure

SSortation DSortation

Δ: DConveyor
φ: Ssortation | DSortation
State: “Sort“
Transition: “t16“
Transition: “t15“
State: “InitialSort“
State: “Init“
Transition: “t100“
State: “FinalSort“
Transition: “t101“
State: “ConvStop“
Transition: “t21“
Transition: “t22“

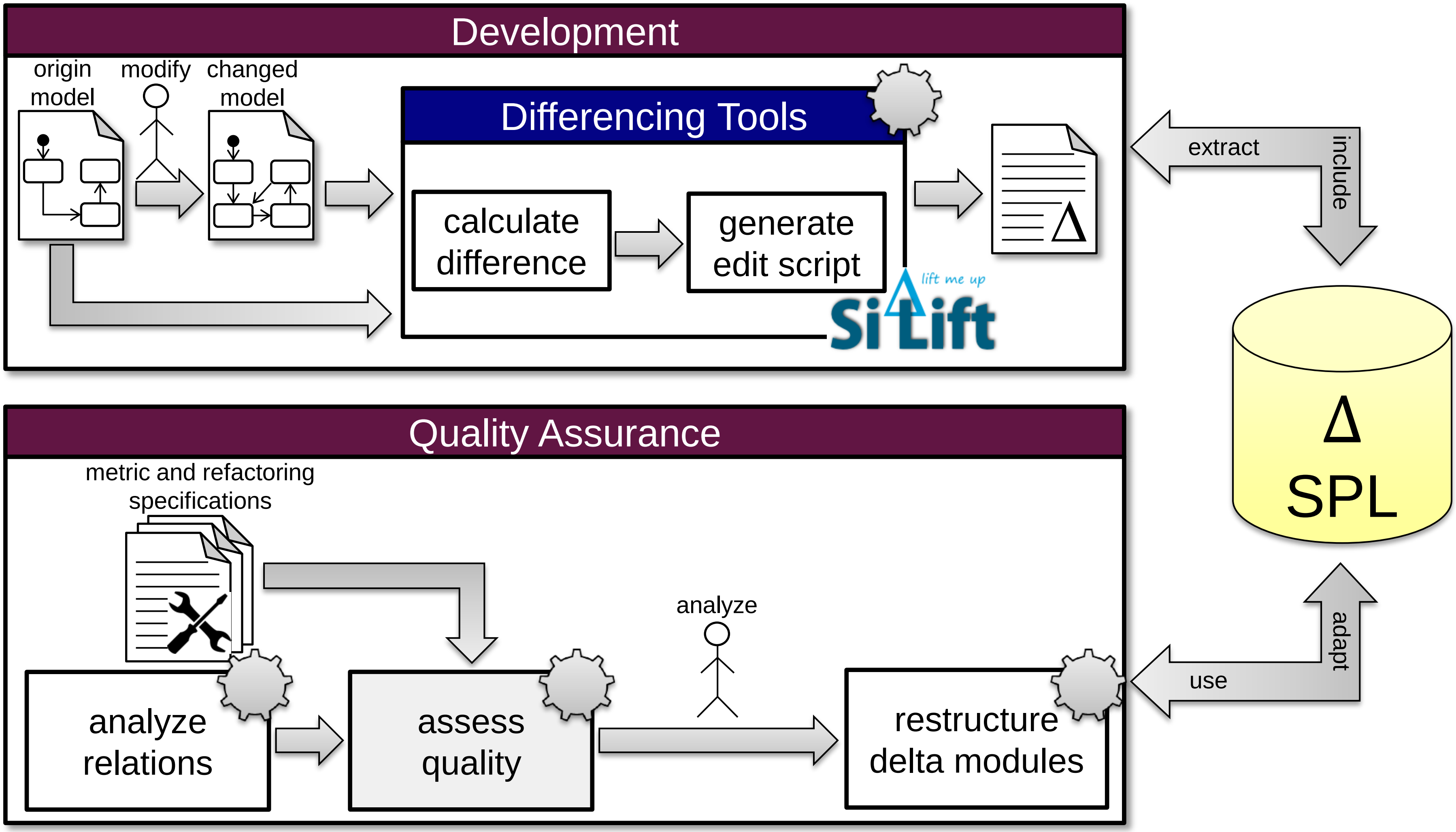
Δ: DSSortation
after DConveyor
φ: SSortation
Transition: “t101“
State: “SSort“
Transition: “t102“
State: “Slide“
Transition: “t103“
Transition: “t104“

Δ: DDSortation
after DConveyor
φ: DSortation

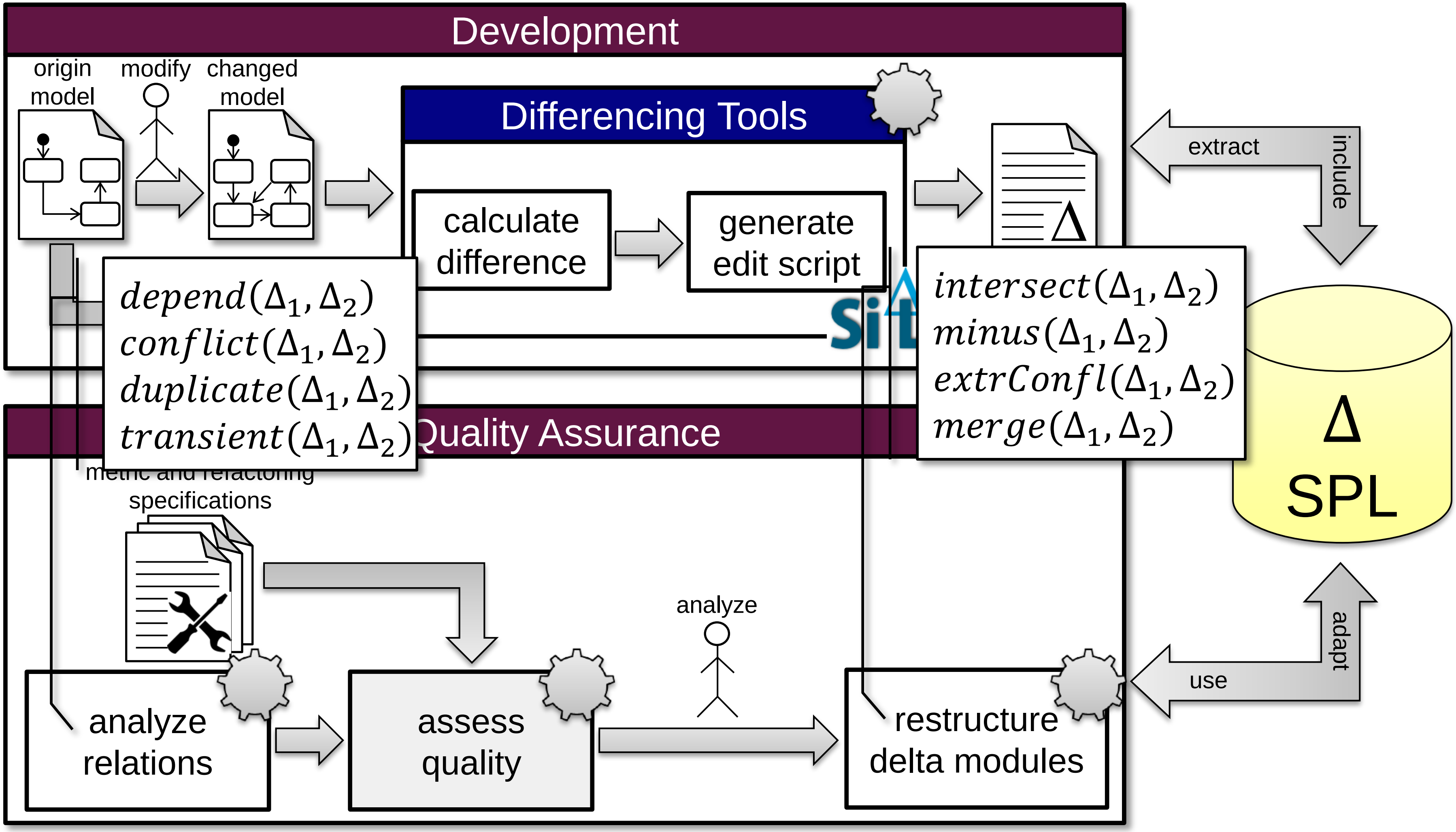
Δ: DSta
φ: Stam
State
Trans
Trans
Trans
State
Trans
Trans

InitialMain
t0
System Behavior
InitialSystemBehavior
t10 t14 t12
Pick Up
Rotate
Put Down
t11 t15 t13
Sort
InitialSort
t100
Init
FinalSort
t102
SSort
t103
t104
Slide
t1
Error
InitialError
t20
Crane Stop
t21
Conv Stop
FinalError
t22
t2
FinalMain

SiPL: A Framework for delta-oriented model-based SPL Engineering



SiPL: A Framework for delta-oriented model-based SPL Engineering



Development

origin model

modify

changed model

Differencing Tools

calculate difference

generate edit script

Δ

$depend(\Delta_1, \Delta_2)$
$conflict(\Delta_1, \Delta_2)$
$duplicate(\Delta_1, \Delta_2)$
$transient(\Delta_1, \Delta_2)$

$intersect(\Delta_1, \Delta_2)$
$minus(\Delta_1, \Delta_2)$
$extrConfl(\Delta_1, \Delta_2)$
$merge(\Delta_1, \Delta_2)$

extract

include

Δ SPL

Quality Assurance

metric and refactoring specifications

analyze relations

assess quality

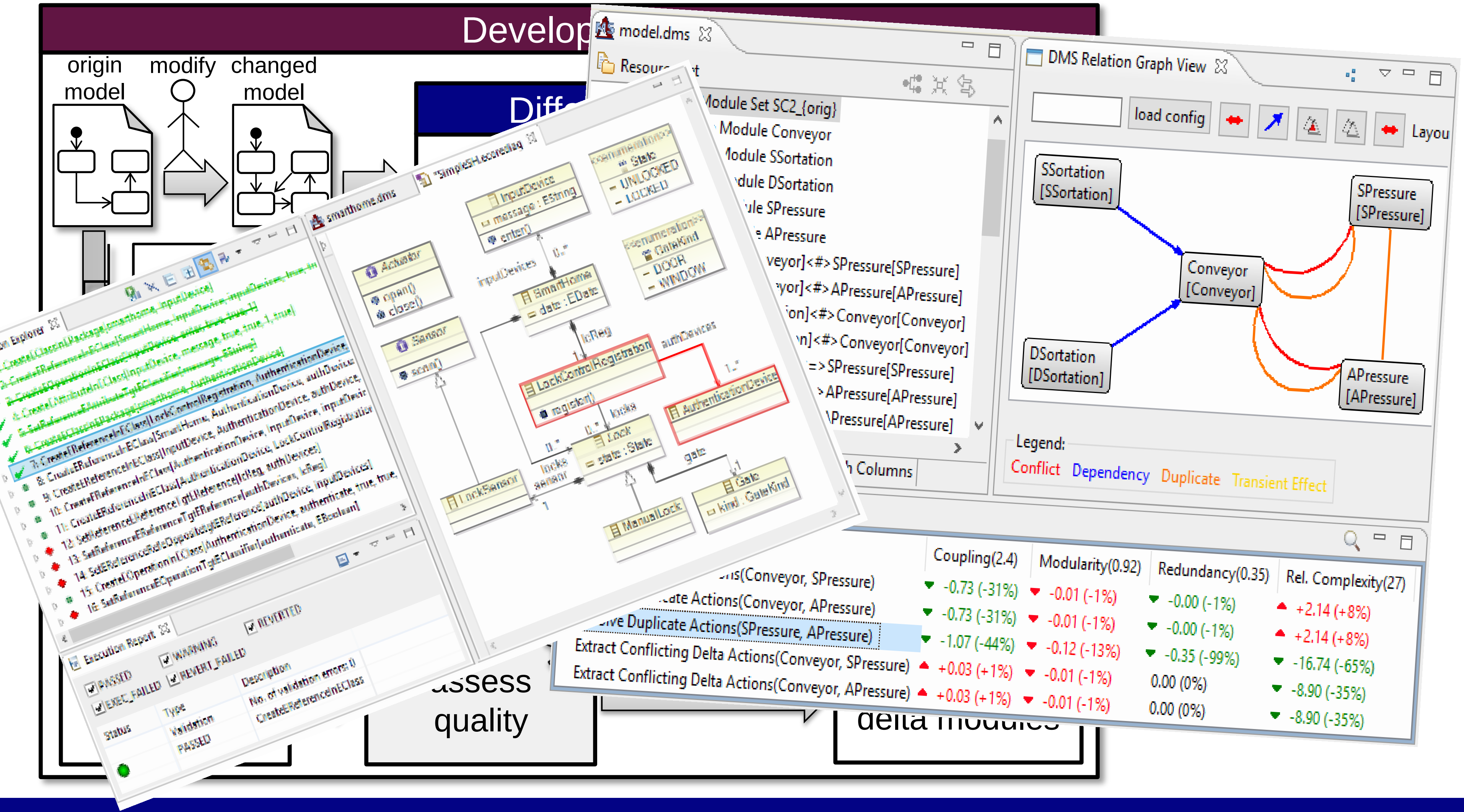
analyze

restructure delta modules

use

adapt

SiPL: A Framework for delta-oriented model-based SPL Engineering



model.dms

Resource Set

- Module Set SC2_{orig}
  - Module Conveyor
  - Module SSortation
  - Module DSortation
  - Module SPressure
  - Module APressure
  - …veyor]<#>SPressure[SPressure]
  - …yor]<#>APressure[APressure]
  - …ion]<#>Conveyor[Conveyor]
  - …n]<#>Conveyor[Conveyor]
  - …=>SPressure[SPressure]
  - …>APressure[APressure]
  - …\Pressure[APressure]

…h Columns

DMS Relation Graph View

SSortation [SSortation]

SPressure [SPressure]

Conveyor [Conveyor]

DSortation [DSortation]

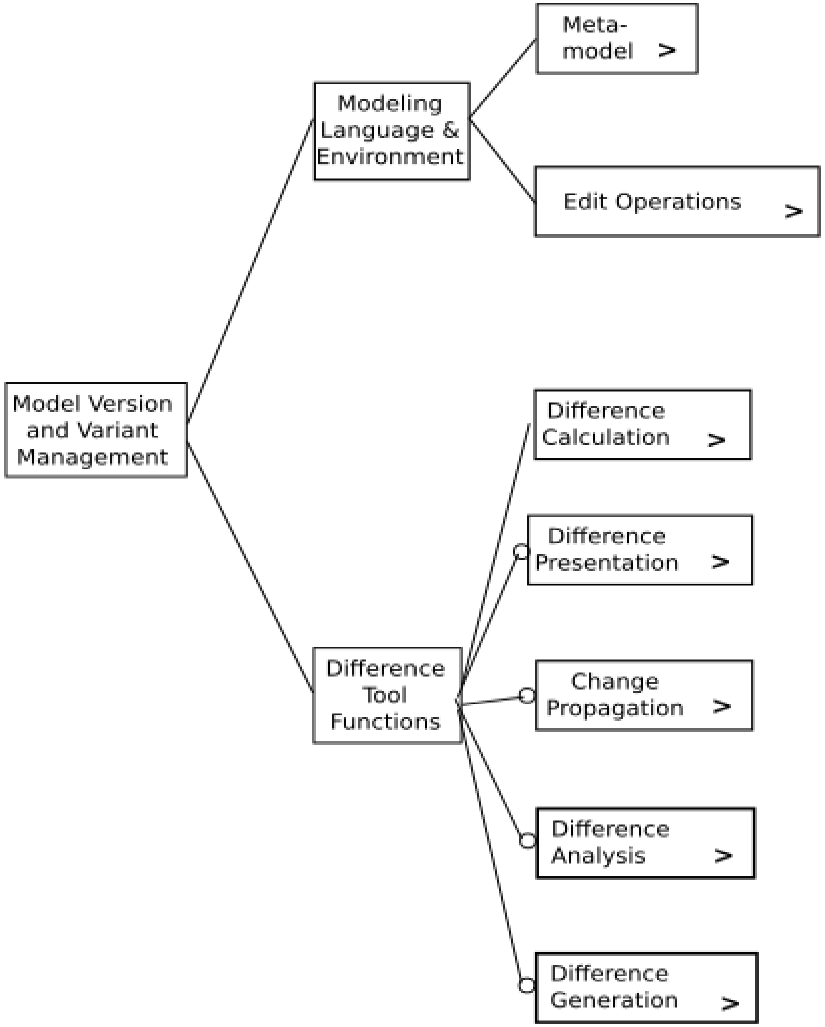
APressure [APressure]

Legend: Conflict Dependency Duplicate Transient Effect

	Coupling(2.4)	Modularity(0.92)	Redundancy(0.35)	Rel. Complexity(27)
…ns(Conveyor, SPressure)	▼ -0.73 (-31%)	▼ -0.01 (-1%)	▼ -0.00 (-1%)	▲ +2.14 (+8%)
…ate Actions(Conveyor, APressure)	▼ -0.73 (-31%)	▼ -0.01 (-1%)	▼ -0.00 (-1%)	▲ +2.14 (+8%)
…ive Duplicate Actions(SPressure, APressure)	▼ -1.07 (-44%)	▼ -0.12 (-13%)	▼ -0.35 (-99%)	▼ -16.74 (-65%)
Extract Conflicting Delta Actions(Conveyor, SPressure)	▲ +0.03 (+1%)	▼ -0.01 (-1%)	0.00 (0%)	▼ -8.90 (-35%)
Extract Conflicting Delta Actions(Conveyor, APressure)	▲ +0.03 (+1%)	▼ -0.01 (-1%)	0.00 (0%)	▼ -8.90 (-35%)

Status	Type	Description
●	Validation PASSED	No. of validation errors: 0

Future Perspective: From Binary to N-ary Operators

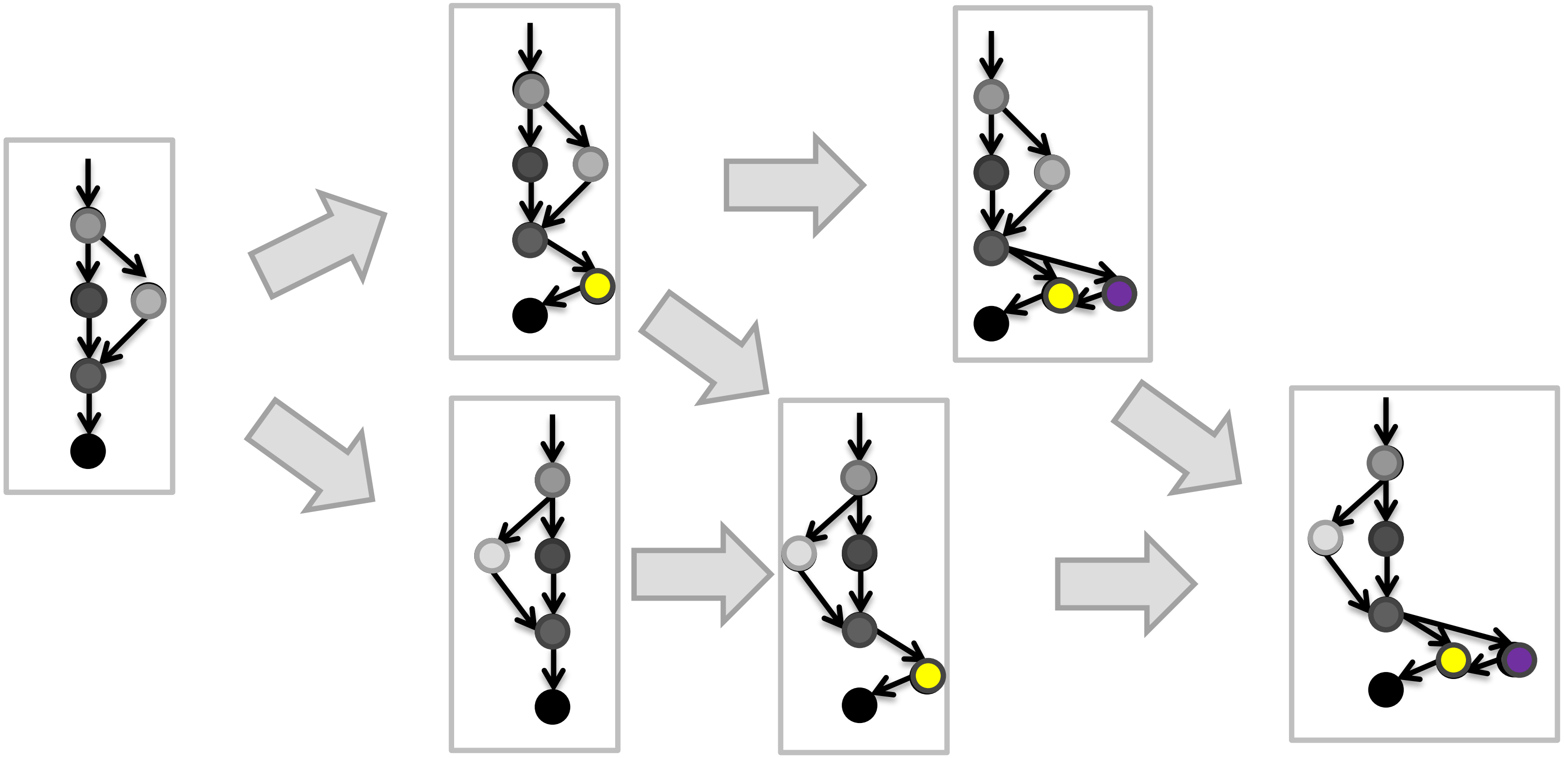

Model Version and Variant Management
Modeling Language & Environment
Meta-model >
Edit Operations >
Difference Tool Functions
Difference Calculation >
Difference Presentation >
Change Propagation >
Difference Analysis >
Difference Generation >

Adaptation to modeling language and environment

Adaptation to use case and application scenario

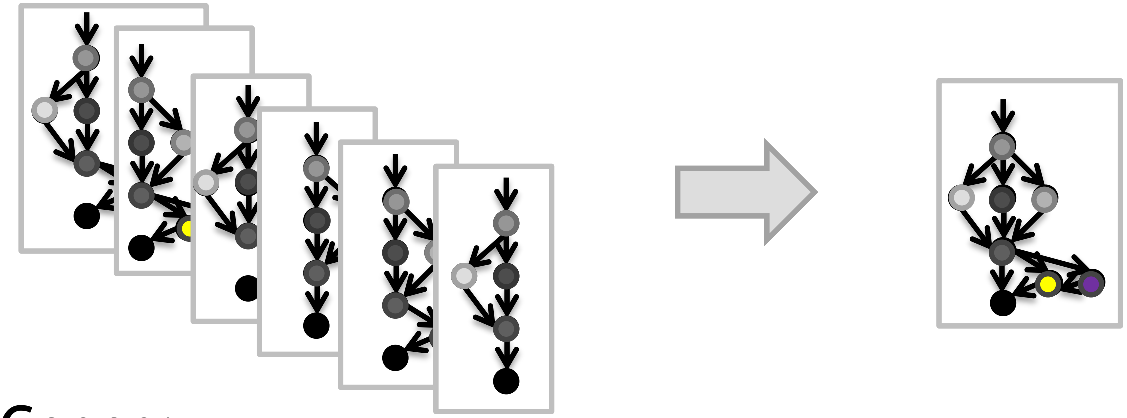
Example: N-Way Matching and Merging

- Context:
  - Parallel development of arbitrary many (autonomous) model variants using “clone-and-own” approach



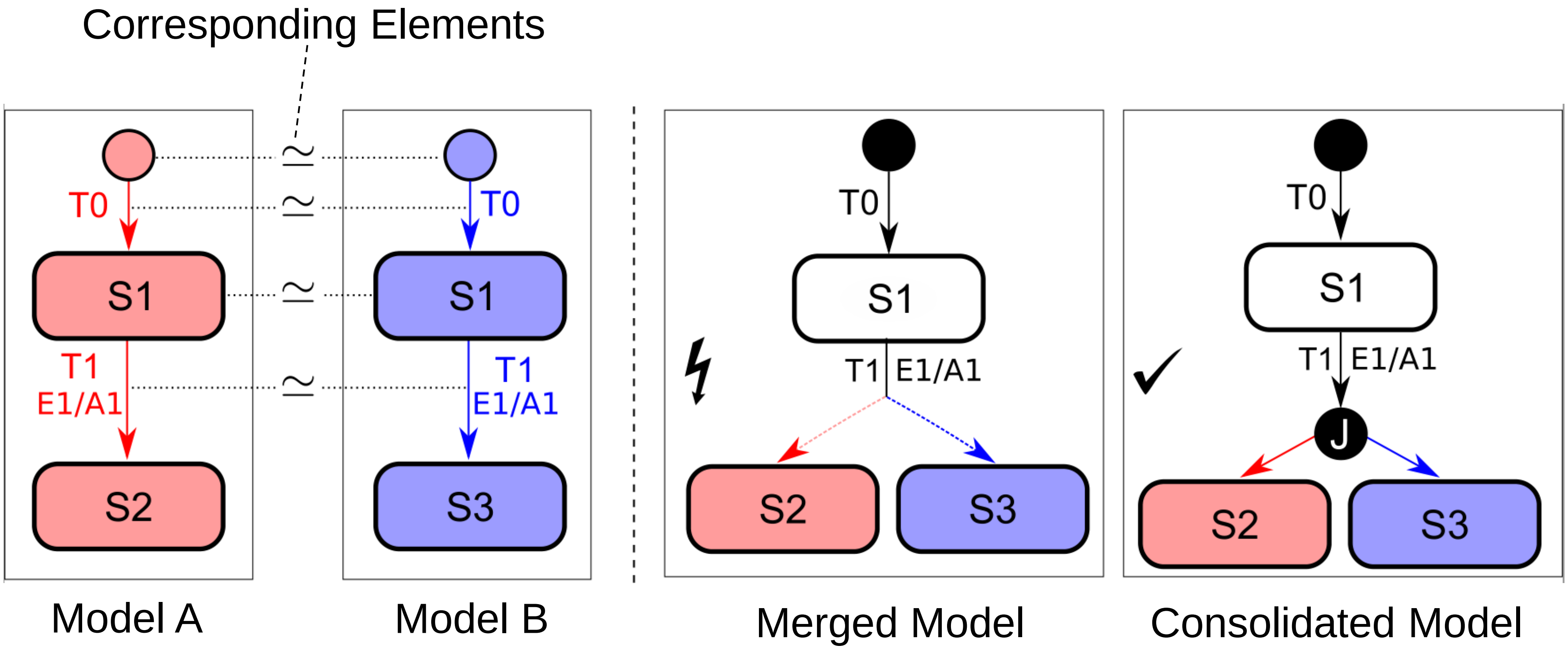
Example: N-Way Matching and Merging

- Goal:
  - Derive family model which integrates (superimposes) all variants into one representation:



- Use Cases:
  - Visualization / Understanding family relations
  - Family-Based Analysis (Testing, Model-Checking,…)
  - Propagate changes (e.g. bug fixes) to all variants at once

Consolidation as a Generalization of Matching and Merging


Corresponding Elements

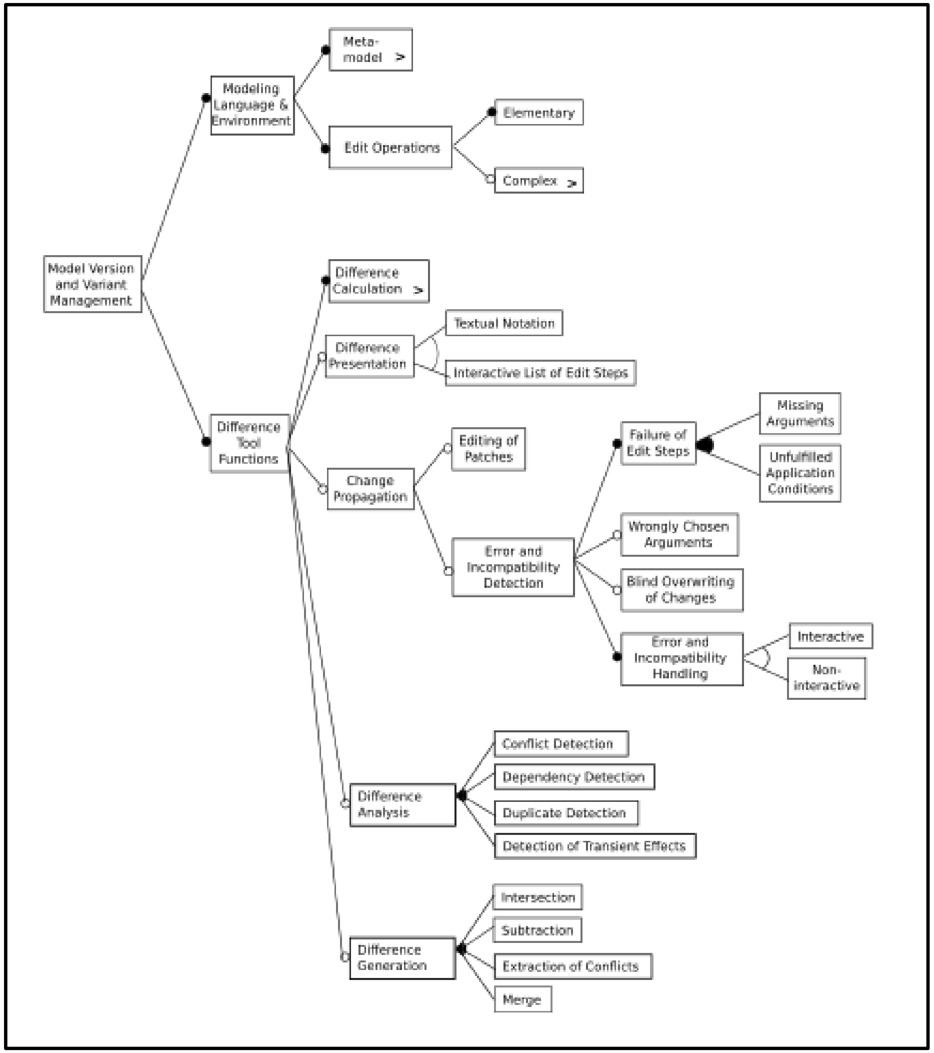
Model A

Model B

Merged Model

Consolidated Model

Summary



- Family of flexible and adaptable versioning operators
- Re-used for several development tasks in the context of managing model variability
- Definitely further re-use potential, e.g.
  - Optimizing change logs obtained from Delta Recorder
  - Supporting variability model evolution in DarwinSPL
  - Optimizing Model-based delta module generation
  - …
- Ongoing work on generalization of operations, e.g. from binary to n-ary

Some Pointers

- Pietsch, C., Reuling, D., Kelter, U. and Kehrer, T., 2017, February. A tool environment for quality assurance of delta-oriented model-based SPLs. In Proceedings of the Eleventh International Workshop on Variability Modelling of Software-intensive Systems (pp. 84-91). ACM.
- Bürdek, J., Kehrer, T., Lochau, M., Reuling, D., Kelter, U. and Schürr, A., 2016. Reasoning about product-line evolution using complex feature model differences. Automated Software Engineering, 23(4), pp.687-733.
- Kehrer, T., Taentzer, G., Rindt, M. and Kelter, U., 2016, July. Automatically deriving the specification of model editing operations from meta-models. In International Conference on Theory and Practice of Model Transformations (pp. 173-188). Springer International Publishing.
- Pietsch, C., Kehrer, T., Kelter, U., Reuling, D. and Ohrndorf, M., 2015, November. SiPL--A Delta-Based Modeling Framework for Software Product Line Engineering. In Automated Software Engineering (ASE), 2015 30th IEEE/ACM International Conference on (pp. 852-857). IEEE.
- Kehrer, T., Kelter, U. and Taentzer, G., 2014. Propagation of software model changes in the context of industrial plant automation. at-Automatisierungstechnik, 62(11), pp.803-814.
- Kehrer, T., Kelter, U. and Reuling, D., 2014, September. Workspace updates of visual models. In Proceedings of the 29th ACM/IEEE international conference on Automated software engineering (pp. 827-830). ACM.
- Kehrer, T., Kelter, U. and Taentzer, G., 2013, November. Consistency-preserving edit scripts in model versioning. In Automated Software Engineering (ASE), 2013 IEEE/ACM 28th International Conference on (pp. 191-201). IEEE.
- Kehrer, T., Kelter, U., Pietsch, P. and Schmidt, M., 2012, September. Adaptability of model comparison tools. In Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering (pp. 306-309). ACM.
- Kehrer, T., Kelter, U. and Taentzer, G., 2011, November. A rule-based approach to the semantic lifting of model differences in the context of model versioning. In Proceedings of the 2011 26th IEEE/ACM International Conference on Automated Software Engineering (pp. 163-172). IEEE Computer Society.