Model-versioning-in-the-large: Algebraic foundations and the tile notation

*Work in progress*

Zinovy Diskin, Krzysztof Czarnecki and Michal Antkiewicz

*Generative Software Development Lab*
*University of Waterloo, Canada*
A “very large” picture:

- Pictures/diagrams are good but informal (no formal semantics)
- Formulas are bad (mind boggling) but precise
- The best of the two worlds: precise diagrams with formal semantics
Versioning: small vs. large

Plan of the talk:
1. Inside a tile
2. Tile composition
3. Reconciliation
4. Sample “large” scenario
5. Summary/discussion

Our setting: 

2D and Large
Example: Relational tile

Green refers to heuristic activity.

Frame denotes elements’ IDs.

Ann’s model, A

P:Person
name: Jo
pho: 111
age: 30

Bob’s model, B

Q:Person
nickname: Jo
pho: 222

model A’

P’:Person
name: Jo
mPho: 111

model B’

Q’:Person
name: Jon
mPho: 222

Green refers to heuristic activity.
Versioning-in-the-small: Deltas are tiles

A tile as a revision of match

Blue elements are derived (computed)

A tile as a revision of update
Special cases: Idle/identity tiles

(a1) Vertical identity

(b1) Horizontal identity

(ab) $1^A = 1_A$
Metamodel of tiles

Node (Model)

Horizontal Arrow (Match spec)

Vertical Arrow (Update spec)

Square (Tile)

\[ \partial_0 \text{ (src model)} \quad \partial_1 \text{ (trg model)} \]

\[ \ell_h \text{ (identical match)} \]

\[ \ell_v \text{ (idle update)} \quad \ell \text{ (idle 2-update)} \]

\[ v\partial_0 \text{ (src update)} \quad v\partial_1 \text{ (trg update)} \]

h\partial_0 \text{ (src match)} \quad h\partial_1 \text{ (trg match)}

Model versioning-in-the-large.
Tile composition & interchange law

\[
AC''_1 = (\tau \otimes \tau') \otimes (\sigma \otimes \sigma')
\]
\[
AC''_2 = (\tau \otimes \sigma) \otimes (\tau' \otimes \sigma')
\]

Interchange law:
\[
AC''_1 = AC''_2
\]
Model versioning-in-the-large. Definition of *tile system* (double category)

- Collection of nodes, vertical and horizontal arrows, and squares (tiles)
- V-arrows can be composed (assoc. and units) and h-arrows can be composed (assoc. and units).
- Tiles can be composed vertically (assoc. and units) and horizontally (assoc. and units), and work together under the interchange law.
Model versioning-in-the-large.
Pasting lemma: Any tile system has the following property:

\[
\{(\tau_{11} \otimes \tau_{12})[\sigma_1 \otimes (\tau_{22} \tau_{32})]\}(\sigma_3 \tau_{33}) = \\
(\tau_{11} \sigma_1) \otimes (\tau_{12} \tau_{22} \tau_{32}) \otimes (\sigma_3 \tau_{33}) = \\
(\tau_{11} \sigma_1) \otimes \{(\tau_{12} \tau_{22}) \otimes \sigma_3\}(\tau_{32} \otimes \tau_{33})
\]
Optimistic versioning and reconciliation

Algebraic laws:

(1) \((\tau \otimes \sigma)! = \tau! \otimes \sigma!\)

(2) \((\tau \odot \tau')! = \tau'!\) (optional)
Use case: “large” versioning scenario via tiles
Summary

- The elementary unit (molecule) of model versioning is a 2D-structure -- tile. Complex scenarios are composed from tiles.
- Tile composition is regulated by algebraic laws of double categories (associativity, interchange law, pasting lemma).
- Complex scenarios are terms built from tiles in some signature of tile operations. Hence,
- Algebraic machineries of category theory become applicable (diagram chasing/diagrammatic calculus).
Even bigger picture

- Engineering (e.g., mechanical and electrical)
- Physics
- Mathematics
- Category theory (abstract nonsense)
  - Higher-dimensional category theory

Software Engineering
Thank you!

Questions/Comments?