Consistency Checking of Conceptual Models via Model Merging

Mehrdad Sabetzadeh (U Toronto, Canada)
Shiva Nejati (U Toronto, Canada)
Sotirios Liaskos (York U, Canada)
Steve Easterbrook (U Toronto, Canada)
Marsha Chechik (U Toronto, Canada)

International Requirements Engineering Conference (RE’07)
October 19, 2007
Collaborative RE
Distributed Development

- Different teams work on separate models
  - Models independent but related
  - Relationships between models described explicitly
Distributed Development

Different teams work on separate models

- Models independent but related
- Relationships between models described explicitly

M1

StaffMember

Nurse  Doctor

M2

MedTeamMember

Physician  Technician
Different teams work on separate models

- Models independent but related
- Relationships between models described explicitly

Distributed Development
Building Relationships

→ Never entirely sure how concepts are related
Building Relationships

→ Never entirely sure how concepts are related

Calendar
Unit

Schedule
Ward

Relationships need to be validated
Means for Analyzing Relationships

→ Human-centric
  ➔ negotiation, repertory grid, etc.

→ Automatic
  ➔ consistency checking, simulation, etc.
Means for Analyzing Relationships

→ Human-centric
  ➡ negotiation, repertory grid, etc.

→ Automatic
  ➡ consistency checking, simulation, etc.

Focus of this work
Checking Consistency of Relationships: Examples
Checking Consistency of Relationships: Examples
Checking Consistency of Relationships: Examples
Checking Consistency of Relationships: Examples

M1

Widget

↓

Canvas

M2

Widget

↓

Component

↓

Canvas

Cyclic Inheritance
Challenge

→ Often faced with a system of related models

Is the “system” consistent?

➤ Consistency of individual mappings not enough!

➤ ... it does not guarantee global consistency [Van Lamsweerde et al, Nuseibeh et al]
Global Inconsistency: Examples

M1

Line

Rect

M3

Box

Square

M2

Polygon

Rectangle
Global Inconsistency: Examples

Individually Consistent
Global Inconsistency: Examples

Globally Inconsistent
Global Inconsistency: Examples

M1

Widget

Canvas

M2

UIObject → UIContext

M3

Container

Panel
Global Inconsistency: Examples

M1: Widget ➔ Canvas

M2: UIObject ➔ UIContext

M3: Container ➔ Panel

Individually Consistent
Global Inconsistency: Examples

M1

![Diagram showing UIObject and UIContext relationships with a cross indicating globally inconsistent example.]

M2

Global Inconsistency: Examples

M3

Globally Inconsistent
Existing Research

\[ \exists x, y, z, t \cdot R(x, y) \land R(z, t) \land Reach(x, z) \land Reach(t, y) \]
\[ \exists x, y, z, t \cdot R(x, y) \land R(z, t) \land \\
Reach(x, z) \land Reach(t, y) \]
Existing Research

∃x, y, z, t · \( R(x, y) \land R(z, t) \land Reach(x, z) \land Reach(t, y) \)

Rule coupled with mapping!

Not generalizable to global consistency checking!
Our Solution

Construct a merge *before* checking consistency!
Our Solution

Construct a merge before checking consistency!

Input Models and Mappings
Our Solution

Construct a merge before checking consistency!

Input Models and Mappings

Merged Model

Traceability Data
Our Solution

Construct a merge before checking consistency!

Input Models and Mappings

Merged Model

Consistency Checking

Traceability Data

Diagnostics
Our Solution

Construct a merge before checking consistency!
Our Solution

Construct a merge **before** checking consistency!

**Key Benefit:**

*Can detect global inconsistencies without coupling the rules with the mappings!*

---

**Inconsistency Projection**
Approach in Action

M1

Line

Rect

Box

Square

M2

Polygon

Rectangle

M3
Demo:

http://www.cs.toronto.edu/~mehrdad/tremer/re07demo
Core Idea:

Check inter-model constraints via checking intra-model constraints of a merged model
Building Blocks

Model Merging
Consistency Checking Rules
Diagnostics Generation
Model Merging
[ICSE’01, ASE’03, FSE’04, RE’05, ICSE’07]

→ Builds on [RE’05]

→ Highlights
  ➜ Concentrates on conceptual models
  ➜ Customizable to different graph-based notations
  ➜ Tolerates incompleteness and inconsistency
  ➜ Can merge several models at a time
Model Merging
[ICSE'01, ASE'03, FSE'04, RE'05, ICSE'07]

→ Builds on [RE'05]

→ Highlights
  ➤ Concentrates on conceptual models
  ➤ Customizable to different graph-based notations
  ➤ Tolerates incompleteness and inconsistency
  ➤ Can merge several models at a time

Key for global consistency checking
Consistency Rules in Conceptual Modelling: Some Patterns

→ Endpoint Compatibility

→ Multiplicity

→ Reachability
What Logic?

→ (Standard) First Order Logic?
   ➡ Can't express transitive closure
     ➢ hence, doesn't capture the reachability pattern
   ➡ No counting operator
     ➢ leads to complex formulas for the multiplicity pattern

→ Temporal Logic (CTL / LTL)?
   ➡ Unnatural for structural models
   ➡ Limited quantification power and no counting operator
     ➢ can't capture the multiplicity pattern at all
Language for Consistency Checking

-> We use FO + transitive closure + counting

=> Implementation: CrocoPat [Beyer et al]
Language for Consistency Checking

→ We use FO + transitive closure + counting

⇒ Implementation: CrocoPat [Beyer et al]

→ CrocoPat Examples

⇒ Let $E(x, y)$ denote $x \rightarrow y$

🌟 # of predecessors of “A” ?

🌟 nodes reachable from “D”?
Language for Consistency Checking

→ We use FO + transitive closure + counting

➡️ Implementation: CrocoPat [Beyer et al]

→ CrocoPat Examples

➡️ Let $E(x,y)$ denote $x \rightarrow y$

🌟 # of predecessors of “A”? $n := \#(E(x, "A"))$;

🌟 nodes reachable from “D”?
Language for Consistency Checking

→ We use FO + transitive closure + counting
  ➣ Implementation: CrocoPat [Beyer et al]

→ CrocoPat Examples
  ➣ Let $E(x,y)$ denote $x \rightarrow y$

  # of predecessors of “A”?  
  $n := \#(E(x, "A"))$;

  nodes reachable from “D”? 
  $\text{Reach}(x,y) := \text{TC}(E(x,y))$; 
  $\text{FromD}(x) := \text{Reach}("D", x)$;
Inconsistency Diagnostics

- Generated by instrumenting consistency rules

- Example diagnostics:
  - Multiple inheritance

  ![Image of objects with multiple parents]

  Objects with multiple parents:
  - m2/270  (Parents: m2/269, m2/272)

- Cyclic inheritance

  ![Image of objects on cyclic inheritance paths]

  Objects on cyclic inheritance paths (Only one cycle per object is shown):
  - m/256  (Cycle: m/258 → m/257 → self)
  - m/257  (Cycle: m/256 → m/258 → self)
  - m/258  (Cycle: m/257 → m/256 → self)
Inconsistency Diagnostics

→ Generated by instrumenting consistency rules

⇒ Example diagnostics:

➢ Multiple inheritance

FOR n IN MultipleParents(x) {
  Parent0fN(y) :=
  EX(e, Src(e,n) & Tgt(e,y));
  PRINT n,
  "Parents: ",
  Parent0fN(y);
}

➢ Cyclic inheritance

Objects on cyclic inheritance paths (Only one cycle per object is shown):

- m/256  (Cycle: m/258→m/257→self)
- m/257  (Cycle: m/256→m/258→self)
- m/258  (Cycle: m/257→m/256→self)
Inconsistency Diagnostics

- Generated by instrumenting consistency rules

- Example diagnostics:
  - Multiple inheritance
  - Cyclic inheritance

```plaintext
FOR n IN MultipleParents(x) {
    ParentOfN(y) :=
    EX(e, Src(e,n) & Tgt(e,y));
    PRINT n,
    "Parents: ",
    ParentOfN(y);
}
```

Objects with multiple parents:
- m2/270 (Parents: m2/269 m2/272)

Objects on cyclic inheritance paths (Only one cycle per object is shown):
- m/256 (Cycle: m/258 → m/257 → self)
- m/257 (Cycle: m/256 → m/258 → self)
- m/258 (Cycle: m/257 → m/256 → self)
Inconsistency Diagnostics

→ Generated by instrumenting consistency rules

➡ Example diagnostics:

➢ Multiple inheritance

```plaintext
FOR n IN MultipleParents(x) {
    ParentOfN(y) :=
    EX(e, Src(e,n) & Tgt(e,y));
    PRINT n,
    "Parents: ",
    ParentOfN(y);
}
```

➢ Cyclic inheritance

```plaintext
Objects on cyclic inheritance paths (Only one cycle per object is shown):

- m/256 (Cycle: m/258 → m/257 → self)
- m/257 (Cycle: m/256 → m/258 → self)
- m/258 (Cycle: m/257 → m/256 → self)
```
Inconsistency Diagnostics

- Generated by instrumenting consistency rules
- Example diagnostics:
  - Multiple inheritance
  ```
  FOR n IN MultipleParents(x) {
    Parent0fN(y) :=
    EX(e, Src(e,n) & Tgt(e,y));
    PRINT n,
    "Parents: ",
    Parent0fN(y);
  }
  ```
  - Cyclic inheritance
  ```
  Objects on cyclic inheritance paths (Only one cycle per object is shown):
  • m/256  (Cycle: m/258→m/257→self)
  • m/257  (Cycle: m/256→m/258→self)
  • m/258  (Cycle: m/257→m/256→self)
  ```
Preliminary Evaluation

Performance

Case Study
Performance

![Graph showing performance for different types of inheritance and number of elements.]

- **Dangling Edges**
- **Parallel Edges**
- **Multiple Inheritance**
- **Cyclic Inheritance**

The graph indicates the time (in seconds) required for various operations as the number of elements increases. The performance metrics are categorized into four types: Dangling Edges, Parallel Edges, Multiple Inheritance, and Cyclic Inheritance.
Case Study

Goal: Study the practical utility of global checking

Models: 5 domain models for a healthcare system
  ➔ developed independently by 5 individual students
  ➔ models small (50-70 elements) but realistic

Experiment
  1. Build preliminary relationships between models
  2. Apply global consistency checking to improve these relationships
Observations

→ Global checking ...

⇒ ... useful for exploring design conflicts

StaffMember
belongs to
Schedule

StaffMember
has
Schedule

StaffMember

M1
M2
M3

StaffMember
has
Schedule

M2
M3

StaffMember

Schedule

Merge
Observations

➔ Global checking …

➔ … useful for exploring design conflicts

➔ … collapses several pairwise checks into a single check
Observations

→ Global checking ...  
   ➜ ... useful for exploring design conflicts

StaffMember
belongs to
Schedule

M1

StaffMember
has
Schedule

M2

StaffMember

M3

→ ... collapses several pairwise checks into a single check

Schedule
Schedule
Schedule

M1
M2
M3

Schedule Schedule Schedule Schedule

Merge

→ ... useful for hypothetical reasoning

➢ Example question: What happens if I delete node X?
Tool Support

➜ TReMer+

⇒ Extended version of TReMer

⇒ Tool for Relationship-Driven Model Merging

⇒ New features:

⇒ Structural consistency checking

⇒ for UML domain models, ERD’s, and hierarchical state machines

⇒ Traceability link generation and navigation

⇒ Usability enhancements

http://www.cs.toronto.edu/~mehrdad/tremer/
## Related Work

### Consistency Checking as Model Checking

- **Input:** a model $M$ and a set of properties $P$
- **$M$ inconsistent if $M \not\models P$**

- **Examples**
  - Temporal property checking, e.g. Spin [Holzmann]
  - FO-based constraint checking, e.g. OCL [OMG], xlinkit [Nentwich et al]

### Consistency Checking as Model Finding

- **Input:** a set of properties $P$
- **$P$ inconsistent if there is no $M$ s.t. $M \models P$**

- **Examples**
  - Proving logical consistency in Z [Spivey], Alloy [Jackson], etc.
  - Property-based synthesis [Pnueli]
Related Work

**Consistency Checking as Model Checking**

- Input: a model \( M \) and a set of properties \( P \)
- \( M \) inconsistent if \( M \nmid P \)

**Examples**

- Temporal property checking, e.g. Spin [Holzmann]
- FO-based constraint checking, e.g. OCL [OMG], xlinkit [Nentwich et al]

Our approach falls here!
Related Work

Consistency Checking as Model Checking

- **Input:** a model $M$ and a set of properties $P$

- $M$ inconsistent if $M \not\models P$

**Examples**

- Temporal property checking, e.g. Spin [Holzmann]
- FO-based constraint checking, e.g. OCL [OMG], xlinkit [Nentwich et al]

**Main differences**

- Detection of global inconsistencies
- Built with early RE in mind

Our approach falls here!
Future Work

➡ Handling heterogeneous models
  ➢ ... through metamodel-based transformations and logical model merging

➡ Integration with a model matcher
  ➢ Work in progress! Stay tuned ... 

➡ Resolution of global inconsistencies
Summary

→ Developed a tool-supported technique for global consistency checking
  ➥ Based on model merging

→ Identified patterns for consistency rules in conceptual modelling
  ➥ Compatibility, multiplicity, reachability

→ Did a preliminary evaluation of our work
  ➥ Performance, small case study
Thank You!
Questions?