

A Relationship-Based Approach to Model Integration

Marsha Chechik, Shiva Nejati, Mehrdad Sabetzadeh
*Journal of Innovations in Systems and Software
Engineering, 2012*

Presented by Wentao He

Contents

- I. Introduction
- II. Model Integration Operators
- III. Model Merging Frameworks
- IV. Two Example Merge Operators
- V. Tool Support
- VI. Discussion and Conclusion

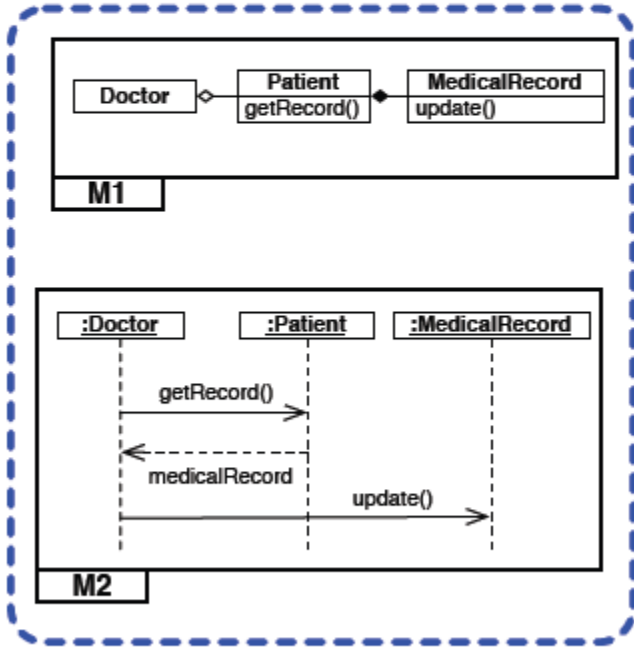
Introduction

Problem:

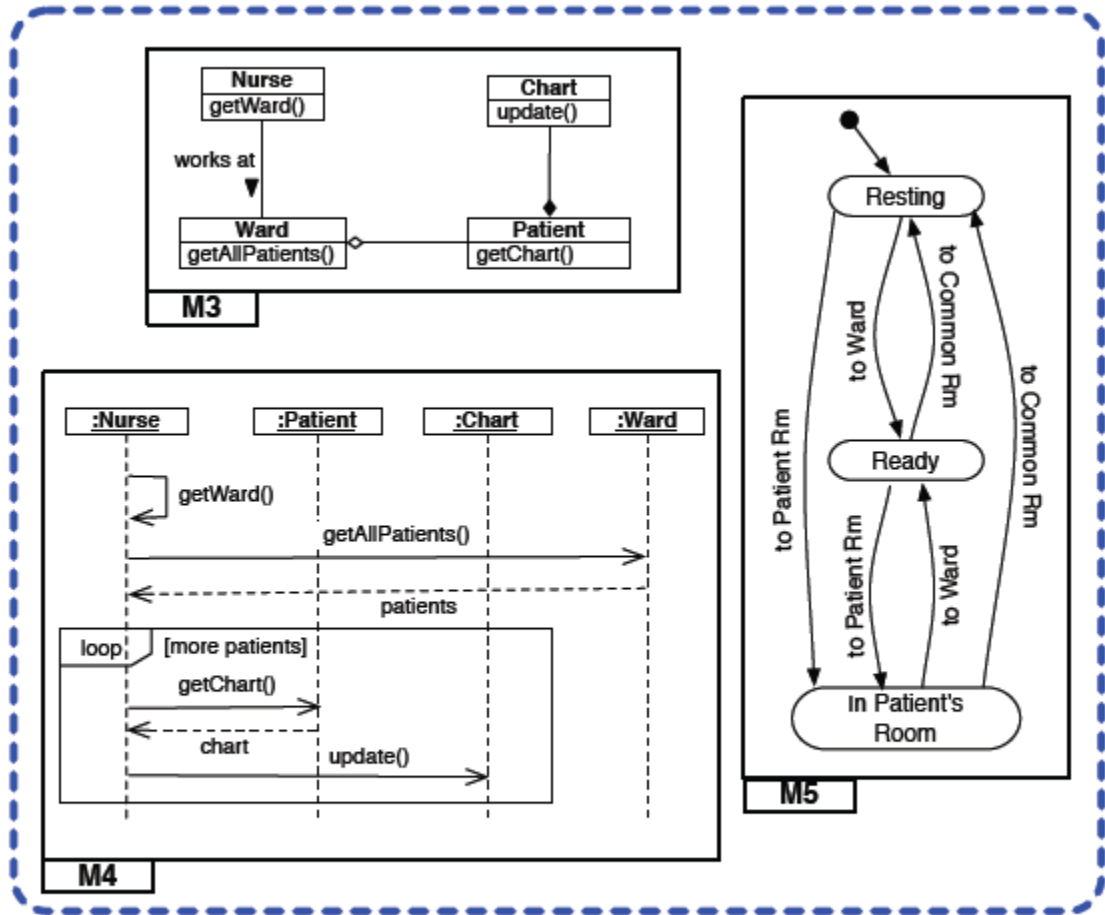
During large-scale development, engineers inevitably have to deal with large collections of models representing different perspectives, different versions across time, different system components, etc.

A key problem is how to integrate these models!

Hospital Information System (HIS) – 1/2

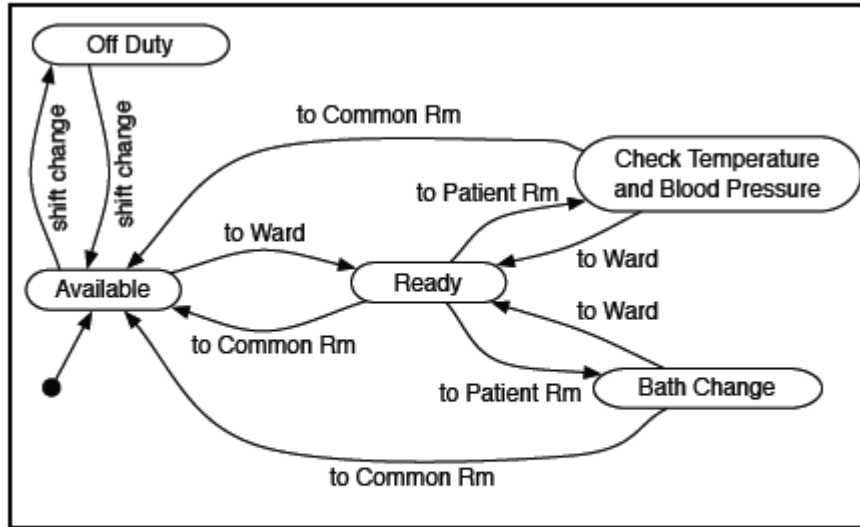


 Doctor



 Nurse I

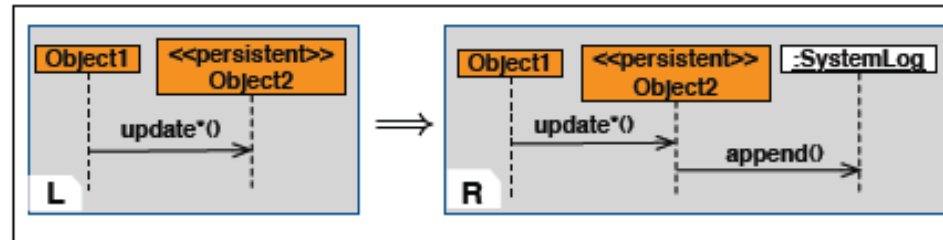
Hospital Information System (HIS) – 2/2



M6



Nurse II



M7



System Administrator

Figure 1. Example models originating from different sources.

Introduction

- Models built for a system (for example: HIS) are not stand-alone objects but inter-related.
- Possible Relationships:
 - models may overlap with one another
 - models may interact at run-time
 - models may be cross-cutting

Introduction

- 3 core types of model integration activities:
 - merge
 - composition
 - weaving

Table 1. Model integration.

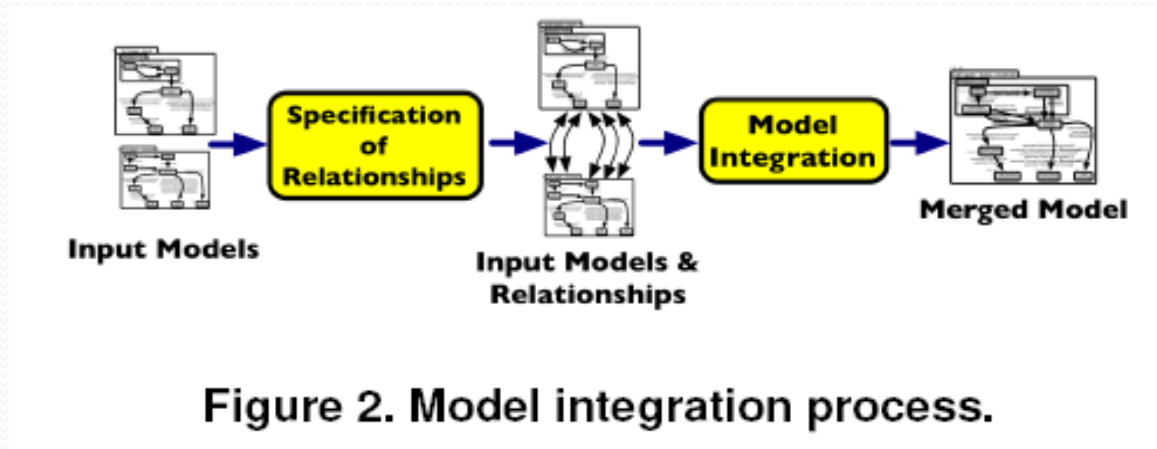
Relationships	Integration operator
Interact	Compose
Cross-cut	Weave
Overlap	Merge

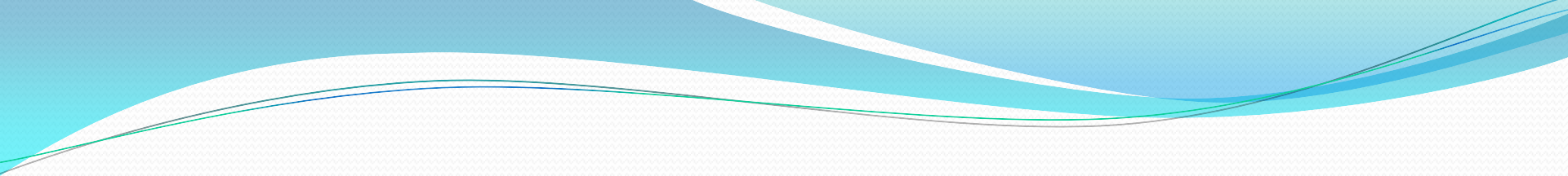
- The choice of integration activity is determined by the relationship between the models. (See Table 1)

Introduction

Model integration process:

- Specify relationships between models
- Choose appropriate operator



- 
- I. Introduction
 - II. Model Integration Operators**
 - III. Model Merging Frameworks
 - IV. Two Example Merge Operators
 - V. Tool Support
 - VI. Discussion and Conclusion

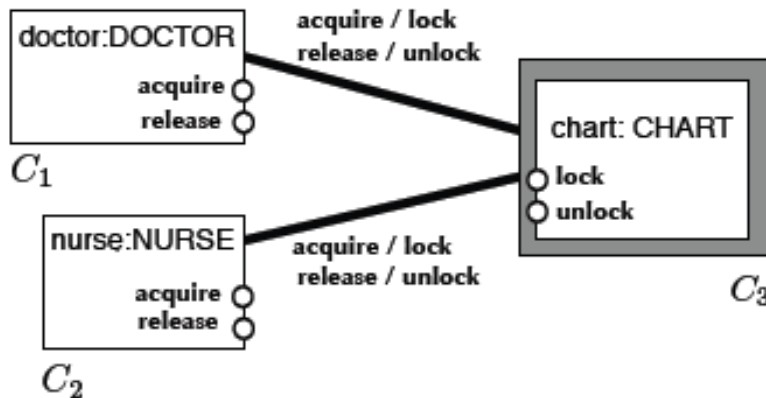
Model Integration Operators

Compose:

- Composition refers to the process of assembling a set of autonomous but interacting models that capture different components of a system.

Model Integration Operators

Compose:



acquire in C₁ maps to *lock* in C₃
release in C₁ maps to *unlock* in C₃
acquire in C₂ maps to *lock* in C₃
release in C₂ maps to *unlock* in C₃

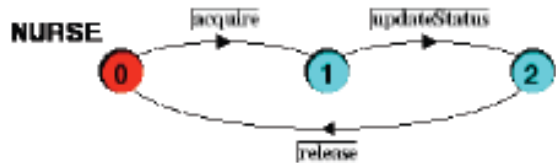
Figure 3. Interconnections between processes running in parallel.

Model Integration Operators

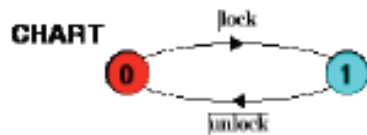
Compose:



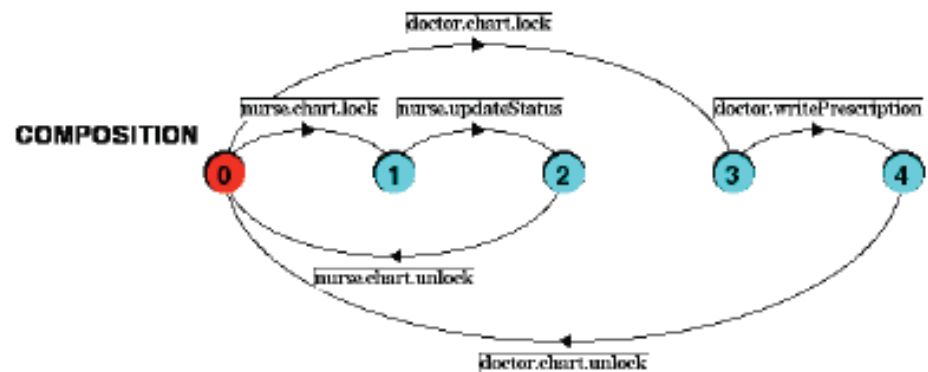
(a)



(b)



(c)



(d)

Figure 4. Specification of processes and their parallel composition.

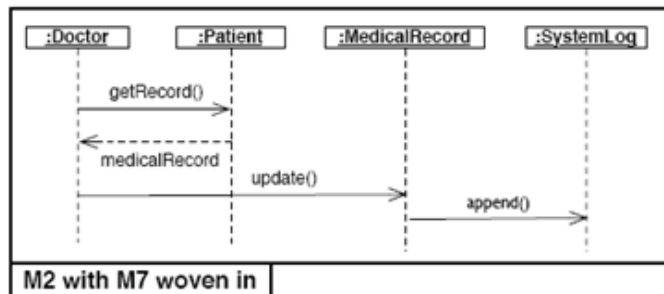
Model Integration Operators

Weave:

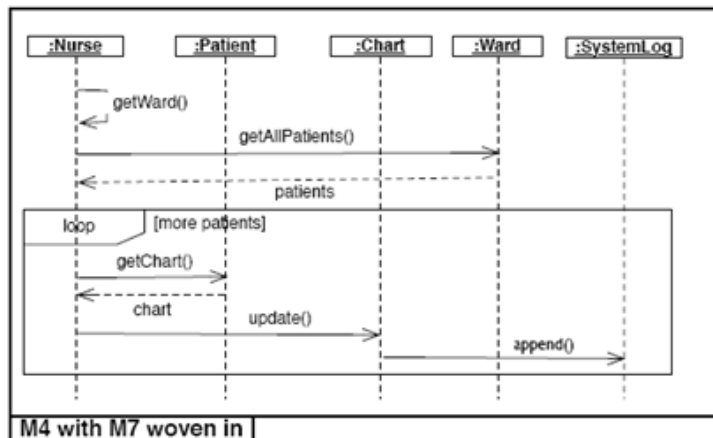
- In Aspect-Oriented Software Development (AOSD), weaving is used to incorporate cross-cutting concerns into a base system.
- A classic example of a cross-cutting concern is logging, affecting all logged activities in a system.

Model Integration Operators

Weave:



Object₁ in M7 maps to Doctor in M2
Object₂ in M7 maps to MedicalRecord in M2
update*() in M7 maps to update() in M2



Object₁ in M7 maps to Nurse in M4
Object₂ in M7 maps to Chart in M4
update*() in M7 maps to update() in M4

Figure 5. Weaving model M7 into M2 and M4.

Model Integration Operators

Merge:

- Merge is used to build a global view of a set of overlapping models that capture different perspectives on a certain functionality.
- The goal of merging is to combine the input models by unifying their overlaps.
- Existing merging approaches:
 - Some require only consistent models be merged
 - Some tolerate inconsistencies

Model Integration Operators

Merge:

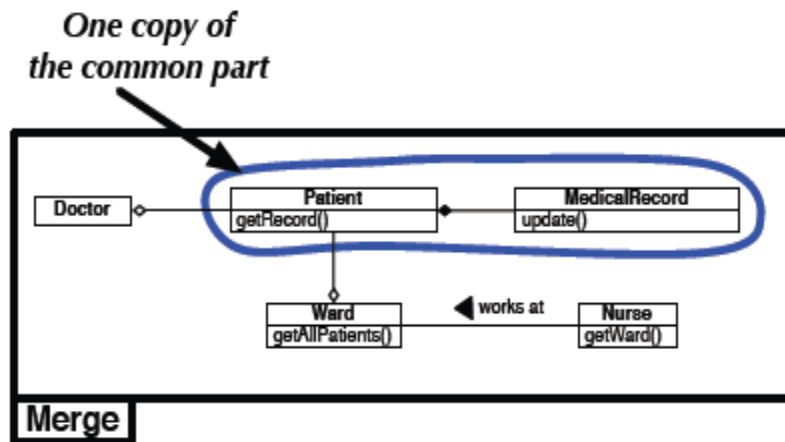


Figure 6. Merge of models M1 and M3.

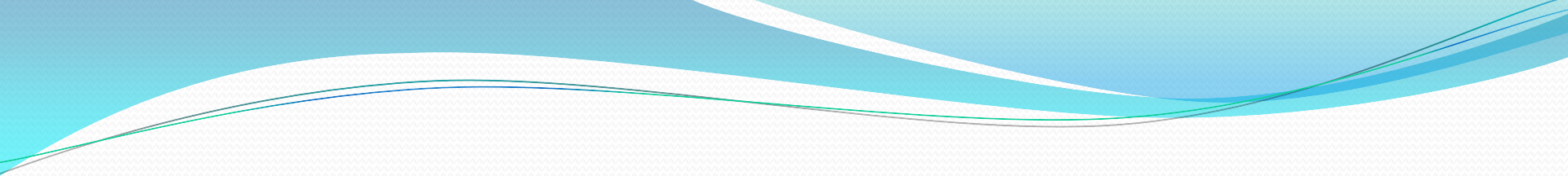
Patient in M1	maps to	Patient in M3
MedicalRecord in M1	maps to	Chart in M3
Aggregation in M1	maps to	Aggregation in M3
update() in M1	maps to	update() in M3
getRecord() in M1	maps to	getChart() in M3

Note: The source models here have different vocabularies, preference is given to terminology in one of the source models.

Model Integration Operators

Important Considerations:

- What notation(s) are the source models expressed in?
- What assumptions are made about the application context, the nature of the models, and their intended use?
- What are the exact details of the relationships that need to be established (in terms of the level of granularity, semantics, representation)?
- What quality and correctness criteria do we expect of the result of the integration?

- 
- I. Introduction
 - II. Model Integration Operators
 - III. Model Merging Frameworks**
 - IV. Two Example Merge Operators
 - V. Tool Support
 - VI. Discussion and Conclusion

Model Merging Frameworks

Overlap Relationships:

- Before a merge can be applied, the exact nature of the overlap of the concepts between the two models needs to be explicated.
- Overlap types:
 - Equivalence
 - Similarity
 - Generalization
 - Aggregation
 - Overriding
 - Information Gaps

Table 2. Different types of overlapping relationships.

	Graph-Based (Class Diagrams)	Semantics-Based (State Machines)
Overlap Type	<p>Equivalence</p> <p>P1 P2 1</p>	<p>P1 P2 2</p>
	<p>Similarity</p> <p>P1 P2 3</p>	<p>P1 P2 4</p>
	<p>Generalization</p> <p>P1 P2 5</p>	<p>P1 P2 6</p>
	<p>Aggregation</p> <p>P1 P2 7</p>	<p>Not Applicable</p> <p>8</p>
	<p>Overriding</p> <p>P1 P2 9</p>	<p>P1 P2 10</p>
	<p>Information Gap</p> <p>P1 P2 P3 11</p>	<p>P1 P2 P3 12</p>

Model Merging Frameworks

Equivalence:

An equivalence mapping between two elements means that they refer to exactly the same concept in the real world.

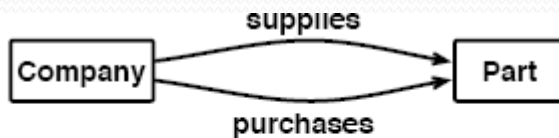
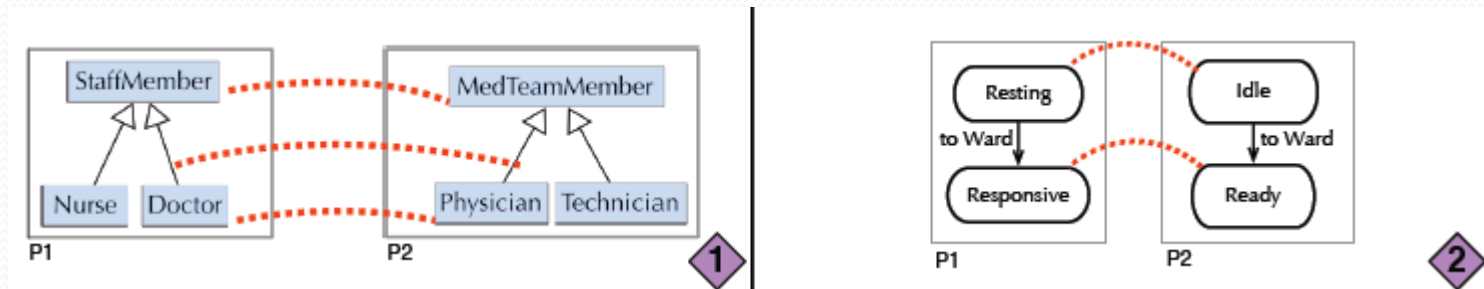


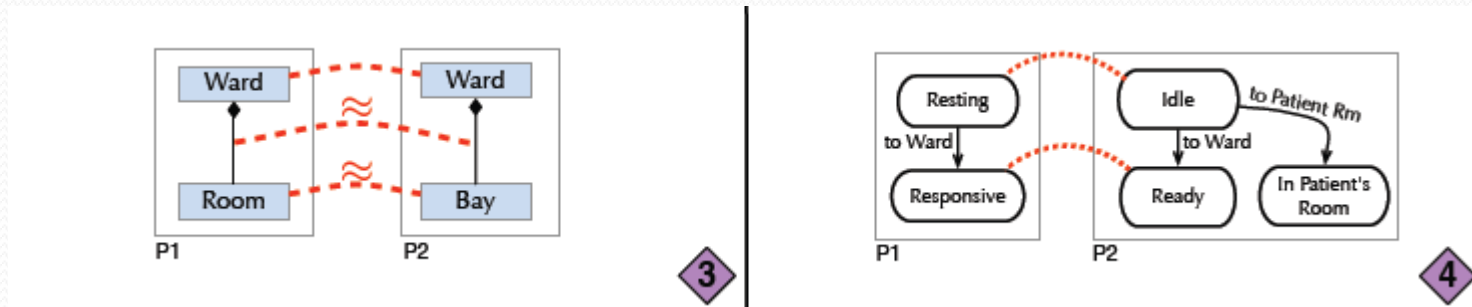
Figure 7. Associations with same endpoints.

Note: Even when the two endpoints are equivalent, links between them may not be so.

Model Merging Frameworks

Similarity:

Two elements might be similar in some respects but not necessarily equivalent.



Model Merging Frameworks

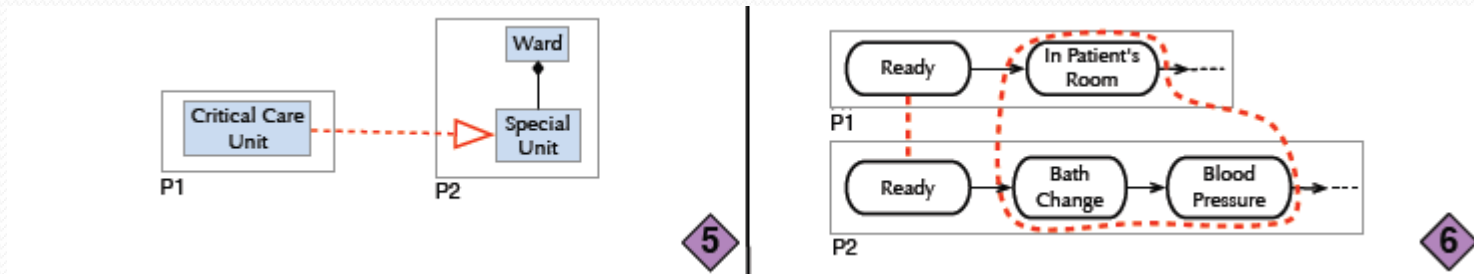
Equivalence vs Similarity:

- Two elements related via equivalence relation will be collapsed into one by merge. If this is intended, use equivalence, otherwise similarity.
- Equivalence is a transitive notion, whereas similarity is not.

Model Merging Frameworks

Generalization:

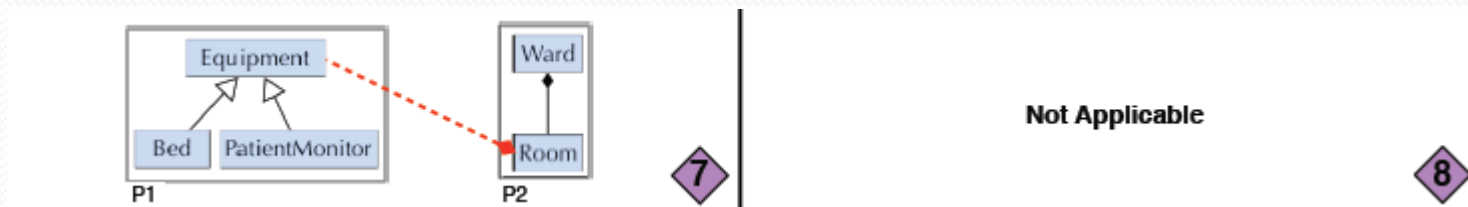
An element in one model can be a generalization or specialization of elements in another model.



Model Merging Frameworks

Aggregation:

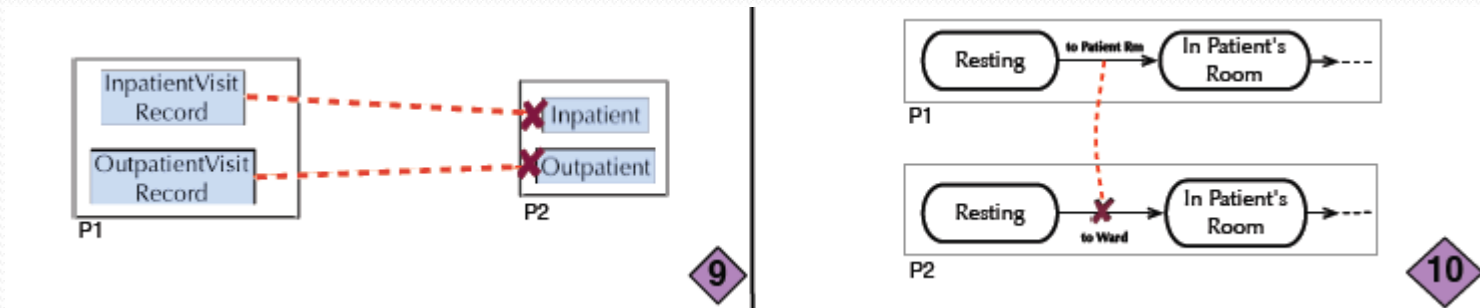
A pair of elements can be related through a has-a or is-part-of relationship.



Model Merging Frameworks

Overriding:

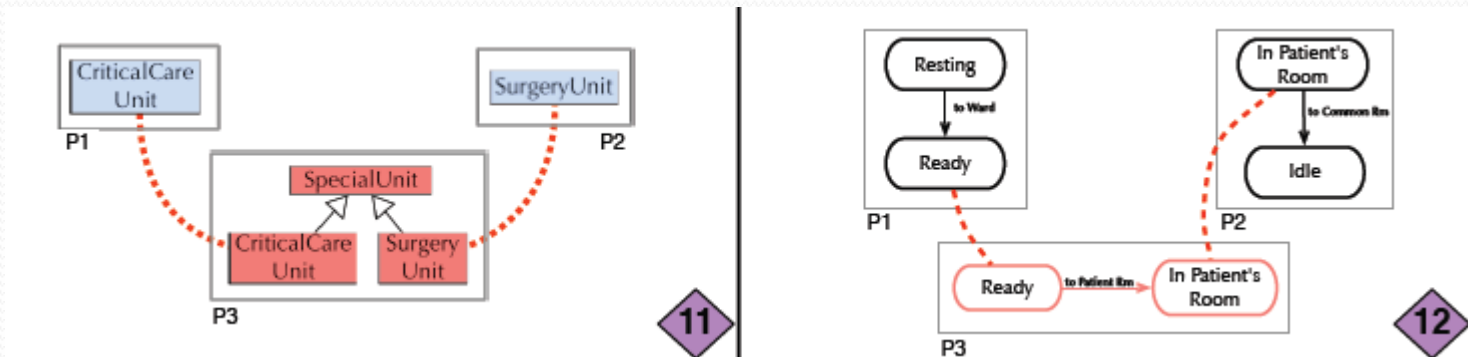
Override, or retrenchment allows developers to withdraw from their positions as their knowledge evolves or to avoid inconsistency with other developers.



Model Merging Frameworks

Information Gaps:

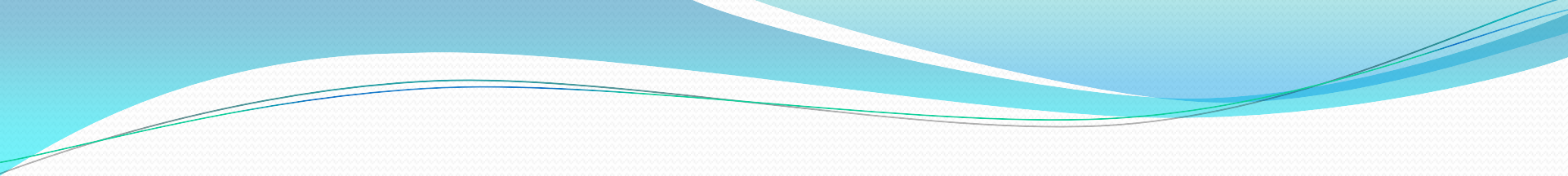
There may be information gaps between the source models that need to be bridged first, before meaningful relationships can be defined.



Model Merging Frameworks

Desirable Criteria for Merge:

- Completeness
- Non-redundancy
- Minimality
- Totality
- Soundness

- 
- I. Introduction
 - II. Model Integration Operators
 - III. Model Merging Frameworks
 - IV. Two Example Merge Operators**
 - V. Tool Support
 - VI. Discussion and Conclusion

Two Example Merge Operators

Two instantiations of merge operator:

- Algebraic merge
- State Machine Merge

Two Example Merge Operators

Algebraic Merge:

- Algebraic merge is a generic operator that works over graph-based models.
- Relationships between the models are captured by sub-graphs, also referred to as connectors.
- The outcome of merge is characterized by an algebraic concept called colimit.

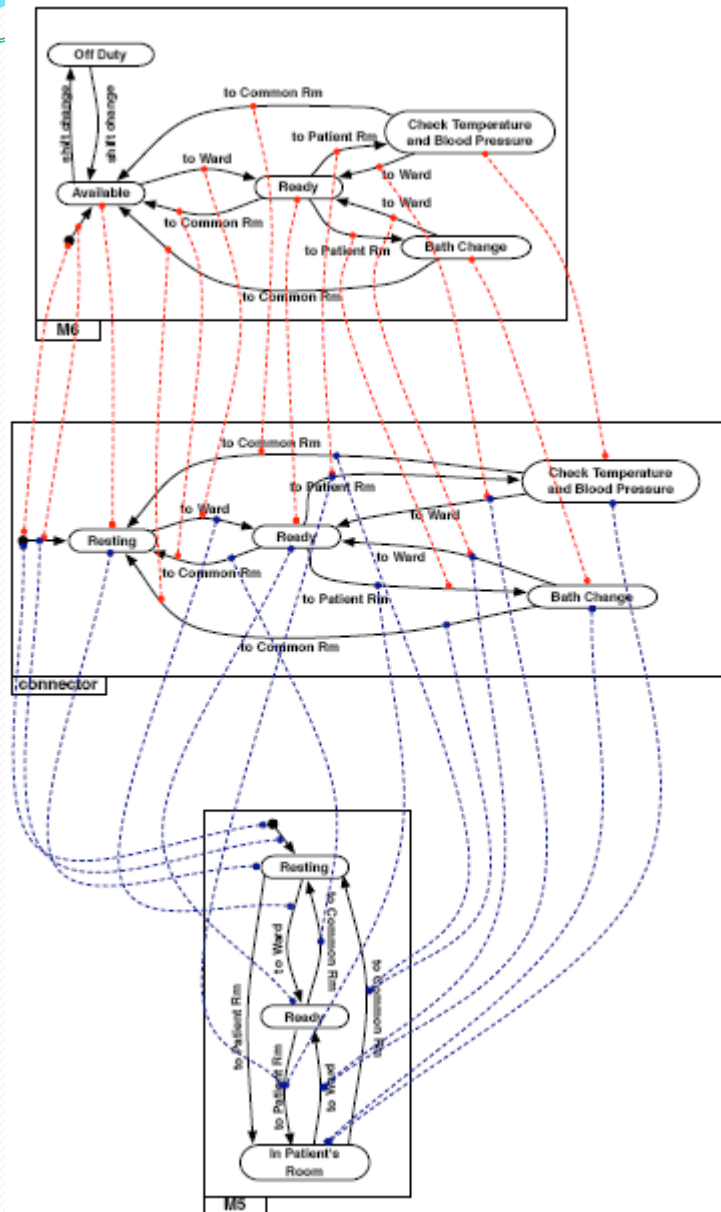


Figure 8. Mappings between the connector model and models M5 and M6.

Example: Algebraic merge of M5 and M6

Step 1: Create a connector model

Step 2: Compute the merge with respect to their overlaps as described by the connector.

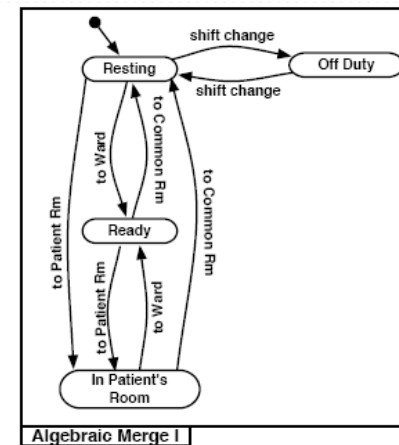


Figure 9. Algebraic merge of state machines M5 and M6 with respect to the mappings in Figure 8.

Note: algebraic merge unifies into a single element any set of source model elements that have been mapped through the connector.

Two Example Merge Operators

State Machine Merge:

- Specifically aimed at state machine models.
- Motivated by the need to preserve the semantic properties of these models in their merges.

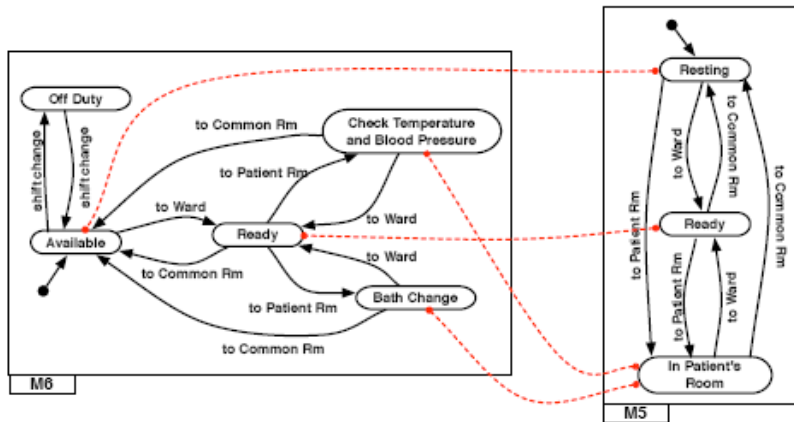


Figure 11. Binary relation between the state machines M5 and M6.

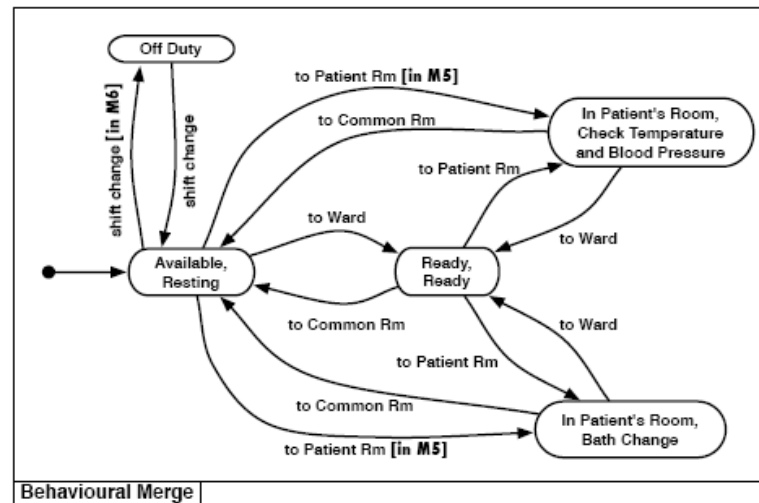


Figure 12. State machine merge of the state machines M5 and M6 with respect to the mapping in Figure 11.

Example: State machine merge of M5 and M6

Step 1: Identify a relationship between the input state machines.

Step 2: Compute the merge with respect to the binary relation defined above.

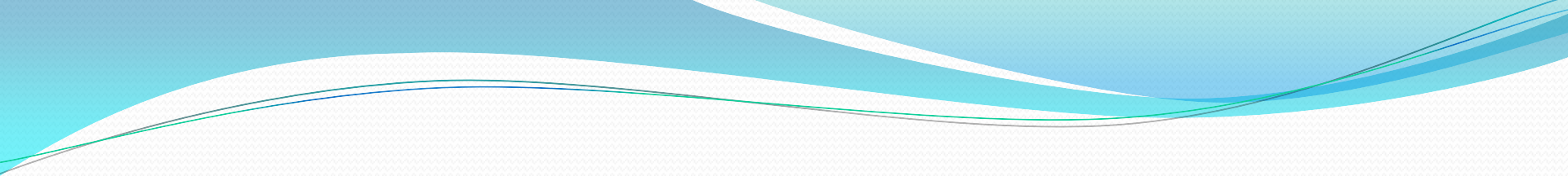
Note:

1. Non-shared behaviors are guarded by conditions;
2. Unlike the algebraic merge, state machine merge does not collapse distinct states.

Two Example Merge Operators

Table 3. Characteristics of algebraic and state machine merge operators.

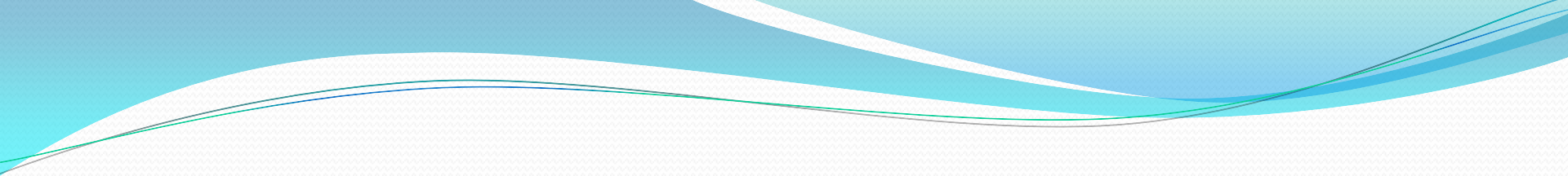
Merge operators	Input Models	Relationships	Soundness	Properties
Algebraic Merge	Homogeneous, graph-based Open-world assumption	Applies to all overlap relations in Table 2	Preserves positive existential LFP properties and one level of universal quantification	Complete, Non-Redundant, Minimal, Total
State Machine Merge	Homogeneous, state machines with trace-based semantics Closed-world assumption	Applies to the overlap relations (2), (4), (6), and (12) in Table 2	Preserves all LTL properties	Complete, Total

- 
- I. Introduction
 - II. Model Integration Operators
 - III. Model Merging Frameworks
 - IV. Two Example Merge Operators
 - V. **Tool Support**
 - VI. Discussion and Conclusion

Tool Support

- The two merge operators described above are implemented as part of a tool called TReMer+.
- TReMer+ has been applied in two real case studies, both dealing with independently-developed models.
- Most recent version of TReMer+ along with material are available at:

<http://se.cs.toronto.edu/index.php/TReMer+>

- 
- I. Introduction
 - II. Model Integration Operators
 - III. Model Merging Frameworks
 - IV. Two Example Merge Operators
 - V. Tool Support
 - VI. **Discussion and Conclusion**

Discussion and Conclusion

- The complexity of integration of a set of models can be reduced by explicating the type of relationships which hold between the models.
- The result of this article would provide a useful guide for the development of new model integration operators.