Concurrent Representations for Jointly-executing Models

Anand Natrajan, Paul F. Reynolds, Jr.

Technical Report No. CS-2001-20
July 27, 2001

Contact: anand@virginia.edu

Web: ftp://ftp.cs.virginia.edu/pub/techreports/CS-2001-20.ps.Z

Concurrent Representations for Jointly-executing Models

Anand Natrajan, Paul F. Reynolds, Jr.
Department of Computer Science, University of Virginia
{anand, reynolds}@virginia.edu

Abstract

Traditionally, when multiple models of a single phenomenon are executed jointly, the representations of all but one model are discarded at any given time. We propose preserving the representations of jointly-executing models at all times. Maintaining concurrent representations can eliminate inconsistencies encountered by previous approaches for joint execution. We present a Multiple Representation Entity (MRE) as a method for maintaining concurrent representations.

1 Introduction

Multiple models of a real-world object or process may be simulated together in order to capture their combined semantics. The joint execution of multiple models, also called Multi-Resolution Modelling (MRM), involves resolving conceptual and representational differences between the models in order to simulate them together. MRM includes but is not restricted to models executed as computer programs or simulations. Joint execution of multiple models enables their re-use, thus avoiding expensive design effort [DAVIS92].

Previous approaches to the MRM problem have been either ineffective, inefficient or both. These approaches, e.g., aggregation-disaggregation and selective viewing, may not achieve consistent joint execution for all models. In selective viewing, only the most detailed model of a phenomenon is executed at all times. In aggregation-disaggregation, only one model, but not necessarily the most detailed one, is executed at any given time. Although these approaches are satisfactory for some models, for others, they encounter problems such as temporal inconsistency and chain disaggregation [—95] [—97B].

Previous MRM approaches do not satisfy two key requirements for joint execution of multiple models: *multi-representation interaction*, i.e., permitting interations at multiple resolution levels at all times, and *multi-representation consistency*, i.e., maintaining consistency among the multiple resolution levels. We define an *effective* MRM approach to be one that satisfies these requirements [—00].

We have concluded that to be effective all resolutions must be represented. Surprisingly, this approach can be cost effective [—97A]. In this paper we discuss the requirements for effective MRM and show how to maintain consistency among multiple models, even when concurrent interactions at different levels of resolution occur.

Preserving the representations of all models and permitting changes to them at all times can lead to effective joint execution of those models. We call model representations that exist and can be changed at all times *concurrent representations*. We show why maintaining concurrent representations for multiple models can lead to effective MRM even when previous approaches fail. We present a Multiple Representation Entity (MRE) as an implementation of concurrent representations. Maintaining consistency among multiple representations when concurrent interactions occur is the key challenge with an MRE. Briefly, we address how this challenge can be met.

2 Model

A *model* captures the semantics of selected concepts, objects and processes in terms of other well-defined concepts, objects and processes. Objects and processes are called *entities* in a model. For example, atoms and molecules may be the entities in a chemical model. A model consists of representation for entities, relationships among the representations and interactions that change the state of the representations. Formal equivalents of this characterisation of a model can be found in modelling methodologies such as Object Modelling Technique [RUM91], Object Oriented Analysis [SHLAER92], Object Model Template [OMT98] and Unified Modelling Language [ALHIR98]. When a model *executes*, it simulates the progress of the phenomenon modelled, implying the passage of time. The representation and relationships in a model may change with time. These changes cause a change in the behaviour of the model.

Representation: The *representation* of an entity is a means of describing the entity and its properties. The representation of a model is the union of the representations of entities. An *attribute* is an element of the representation of an entity that captures a property of the entity. In a chemical model, position, charge, valency and energy may be the attributes of atom and molecule entities.

Relationships: A *relationship* between two attributes indicates how the value of one attribute changes when the value of the other attribute changes. In a valid or consistent model, the relationships among attributes *hold*, i.e, the values of attributes change in accordance with the relationships among them. Examples of relationships in a chemical model would be the bonds between atoms, or fundamental laws that govern the behaviour of the entities, such as Boyle's Law.

Interactions: An *interaction* is a communication between entities. The *effects* of an interaction are the changes caused by the interaction to the attributes of its sender and receivers. Continuing with the example of a chemical model, adding reagents or increasing temperature may be interactions. The effects of these interactions would be changes to the positions or energies of atoms and molecules. An interaction that changes only the relationships in a model will cause the state of the model to change as well because of the changed relationships. We do not differentiate between interactions that change the state and interactions that change the relationships in a model. *Concurrent interactions* are those interactions that occur during overlapping simulation time intervals, i.e., during the same time-step.

Time: At a given instant of time, the values of the attributes and the relationships among the attributes reflect the phenomenon being modelled. The state of a model is a set of well-defined values assigned to attributes. As a model executes, its state and the relationships among attributes may change, although the relationships continue to hold. Although these changes may happen continuously, for most practical executions of models, they happen at discrete times. Accordingly, there exists a sequence of observation times $T = (t_0, t_1, t_2, \ldots)$, such that the state of a model is defined only $\forall t_i \in T$. For $t_j \notin T$, the state of the model may be undefined or may be the same as the state at the time $t_i \in T$ where t_i is the largest instant in T such that $t_i < t_j$. T is monotonically increasing. The interval between two consecutive times is a *time-step*, denoted by $[t_i, t_{i+1}]$, where $t_i, t_{i+1} \in T$. The durations of time-steps in a particular model may vary, i.e., $\forall t_i, t_{i+1}, t_j, t_{j+1} \in T, i \neq j$, it is not guaranteed that $t_{i+1} - t_i = t_{j+1} - t_j$.

Behaviour: The *behaviour of a model* is the sequence of states of that model [ABADI95] [LAM94] [HOP79]. If two models A and B have the same representation, relationships and interactions but their attributes have different sequences of values or the same sequences of values but at different times, then A and B have different behaviours. The sequence of states for an entity is a subset of the sequence of states of a model, i.e., the *behaviour of an entity* is a subset of the behaviour of the model.

3 Multi-model

Jointly-executing models together constitute a *multi-model*. Simple, well-designed models executing jointly may capture all the facets required for a particular study of an object or process without a designer having to construct one model that captures exactly those facets. Given that the multiple models are of the same object or process, entities common to the models must be made consistent. However, making the entities consistent can become a significant problem if the models make different assumptions about the processes, objects, the rate of progress and the accuracy at which the object or process is modelled [—97B]. Inconsistencies among models may undermine the reasons for executing them jointly.

Multi-representation modelling (MRM) is the joint execution of multiple models of the same phenomenon. A multi-model may consist of several models; however, for ease of exposition, we will consider an example multi-model M consisting of two models, A and B. We use the term *representation level* to describe the level of abstraction of a model. If some models are compositions/decompositions or abstractions/refinements of one another, their representation levels are also called *resolution levels* or *resolutions*. An aggregate model is a relatively low-resolution (high-abstraction, low-decomposition) model, whereas a disaggregate model is a relatively high-resolution (low-abstraction, high-decomposition) model. A *Low Resolution Entity* (LRE) is an entity at a relatively high level of abstraction, and a *High Resolution Entity* (HRE) is an entity at a relatively low level of abstraction. In the chemical models example, a model operating at the atomic level would be a disaggregate model comprising HREs, namely, atoms, whereas a model operating at the molecular level would be an aggregate model comprising LREs, namely, molecules. The resolution levels form a hierarchy, with the highest level being the most abstract or most aggregate one, and the lowest level being the most refined or most disaggregate one. *Aggregation* is the composition of a collection of HREs into a single LRE, and *disaggregation* is the decomposition of an LRE into its constituent HREs.

Two important concerns with multi-models are capturing cross-model relationships, and resolving time-step differentials among the constituent models [—97B]. Here, we address the former and assume the latter.

Cross-model Relationships: If A and B represent overlapping sets of objects or processes, their representations, Rep^A and Rep^B, must be correlated. Correlating the representations in a multi-model is called *consistency maintenance*. Rel^{cross} is a set of relationships that must hold for multiple models to be consistent with one another. A cross-model relationship $r \in Rel^{cross}$ is a mapping $r: P \rightarrow Q$, where $P \subseteq Rep^A \wedge Q \subseteq Rep^B \vee P \subseteq Rep^B \wedge Q \subseteq Rep^A$. If $Rel^{cross} = \emptyset$, then A and B are independent of each other because their representations are unrelated. Then, consistency maintenance reduces to ensuring that the individual models are self-consistent.

Compatible Time-Steps: We assume that the time-steps of A and B are compatible. *Compatible time-steps* means that if T^A, T^B and T^M are the sequences of times associated with A, B and M respectively, then A and B are defined for all times in T^M. T^M is constructed by interleaving T^A and T^B. Accordingly, times that are common to both T^A and T^B (albeit labelled differently) are included in T^M only once. If $T^M = T^A \cup T^B$, then A must be

defined for all times in T^B and B must be defined for all times in T^A. If $T^M = T^A \cap T^B$, then A and B are defined for all $t \in T^M$. Figure 1 shows two ways to construct T^M.

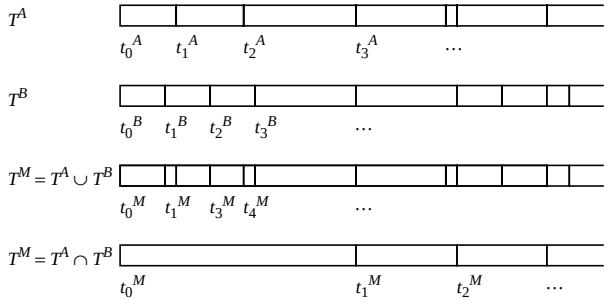

Figure 1: Possible compatible time-steps

4 Requirements for Effective MRM

Any model, including a multi-model, must satisfy its users' requirements. Examples of user requirements are the accuracy of the model, the detail captured by the model and the rate at which the model progresses. The most accurate model of an object or process is the object or process itself; practical models are simplifications that may fail to imitate the object or process in some respects. The Turing test [TURING50] for a model is whether end-users are satisfied that the model captures the facets required for study. A multi-model can satisfy its users' requirements if its constituent models satisfy the users' requirements and the joint execution of the multiple models is effective.

We concentrate on the effectiveness of joint execution of multiple models. Whether an MRM approach is effective or not can be evaluated on the basis of how well it satisfies two requirements:

R1: Multi-representation Interaction: Entities in each model may initiate and receive interactions that may cause changes to the entities concurrently.

The interactions that occur in a multi-model must be those that could occur in any of the constituent models. If an MRM approach satisfies R1, it means the approach does not restrict the execution of any of the models.

R2: Multi-representation Consistency: The representations of jointly-executing models must be consistent with one another. *Temporal consistency* requires that two entities interacting with a third entity at overlapping simulation times have consistent views of the third entity. *Mapping consistency* requires that entity properties common to different models be translated such that repeated translations in a given period do not cause abnormal behaviour in the entity during that period.

R2 is interesting only if the multiple models are related to one another. If $Rel^{cross} = \varnothing$, consistency maintenance and joint execution are uninteresting. Consistent representations are necessary for the consistent behaviour of a multi-model since the state of an entity influences its behaviour [ABADI95] [LAM94] [HOP79].

5 Previous MRM Approaches

Previous MRM approaches can be classified into two broad categories: selective viewing and aggregation-disaggregation. In both these approaches, only one model is executed at any given time. In *selective viewing*, the most detailed model is executed at all times. In *aggregation-disaggregation*, at any given time, depending on the interactions among entities, the system may change the currently-executing model by transitioning among models.

5.1 Selective Viewing

With selective viewing, only the most detailed model is executed, and all other models are emulated by selecting information, or views, from the representation of the most detailed model [DAVIS93]. If A and B model the same object or process, and B is the more detailed model, then B is executed at all times. A may be emulated by selecting information from the representation of B. If T^M is the sequence of times in a multi-model M constructed from A and B, then for all time-steps $[t_i, t_{i+1}]$, where $t_i, t_{i+1} \in T^M$, model B is executed.

Selective viewing is employed when modelling a phenomenon in detail at all times is considered necessary, for example, in some battlefield simulations or computer games. Low-resolution views of a multi-model are generated from the most detailed model. Selective viewing has many disadvantages.

First, executing the most detailed model incurs the highest resource usage cost. Proponents of selective viewing may argue that the smallest detail can affect the execution of the complete model (e.g., a butterfly flapping its wings in Columbia can affect the weather of Western Europe). While this argument may be valid in some cases, for most models, most of the details can be abstracted reasonably in order to conserve resources.

Second, the most detailed model is likely to be the most complex model. One of the main benefits of modelling is to make reasonable simplifications in order to study a phenomenon efficiently. Executing the most detailed model adds complexity instead of reducing it.

Third, executing the most detailed model may limit the opportunities for performing some types of analyses. Abstract models enable a user to make high-level decisions regarding the multi-model. These high-level decisions are likely to change the behaviour of many entities, thus enabling broad analyses of the multi-model. Enabling equivalent analyses in a detailed model requires making corresponding low-level decisions. These low-level decisions may not exist or may be difficult to make. Thus, the equivalent analyses in a detailed model may be impossible or infeasible.

Fourth, some multiple models may not bear hierarchical relationships with one another, i.e., none of them is the most detailed model. Selective viewing implies

that the most detailed model is a monolithic model. For non-hierarchical models, the monolithic model must be created by capturing all the details of all the models. Such a monolithic model requires additional design effort and is likely to be very complex.

The philosophical question of what is the most detailed model can entrap designers into adding ever-increasing detail to a model by refining entities in the model increasingly. However, even assuming a designer can escape this trap eventually, selective viewing is not suitable for the execution of a multi-model because of the above disadvantages.

Selective viewing does not satisfy R1. In selective viewing, interactions that change the representation of the currently-executing model, which happens to be the most detailed one, are permitted at any time, but interactions that change the representation of other models are not permitted. In other words, selective viewing restricts the execution of the other models.

Selective viewing satisfies R2 partially. In selective viewing, cross-model relationships do not hold at all times. Selective viewing forces Rel^{cross} to be null, since only one representation level exists. Forcing cross-model relationships to be null ensures that they hold trivially, but does not capture relationships among jointly-executing models at all observed times.

5.2 Aggregation-Disaggregation

With aggregation-disaggregation, at any given time only one model is executed, although not necessarily the most detailed one. If A and B model the same phenomenon, sometimes only A is executed, and at other times only B is executed. If T^M is the sequence of times in a multi-model M constructed from A and B, then for some time-steps $[t_i, t_{i+1}]$, where $t_i, t_{i+1} \in T^M$, model A is executed, but for other time-steps $[t_j, t_{j+1}]$, where $t_j, t_{j+1} \in T^M$ and $i \neq j$, model B is executed.

Employing aggregation-disaggregation can reduce the resource usage cost of executing a multi-model by executing a low-decomposition constituent model whenever possible. However, aggregation-disaggregation requires transitioning in order to change the currently-executing model. During a transition, the resolution of an entity is changed dynamically to match the resolution of other interacting entities. This dynamic change is called *aggregation* (HREs $\rightarrow$ LRE) or *disaggregation* (LRE $\rightarrow$ HREs). Aggregation-disaggregation ensures that entities interact with one another at the same level by forcibly changing their representation levels. Typically, if an LRE interacts with an HRE, the LRE is disaggregated into its constituents, which interact at the HRE level. LRE-LRE interactions are at the LRE level. A disaggregated LRE may be re-aggregated so that it can interact subsequently at the LRE level. Transitioning can become resource-intensive when it introduces problems such as chain disaggregation, transition latency, network flooding and thrashing [—97B]. However, transitions are necessary in order to avoid inconsistencies that can arise if an LRE interacts with an HRE.

Aggregation-disaggregation does not satisfy R1. In aggregation-disaggregation, in each time-step, either interactions in one model or another are permitted, but not interactions in multiple models. Interactions that change the representation of the currently-executing model are permitted at any time, but interactions that change the representation of other models are not permitted. Although aggregation-disaggregation permits interactions at different representation levels, it nevertheless restricts the execution of some models at a given time.

Aggregation-disaggregation does not satisfy R2. Aggregation-disaggregation forces Rel^{cross} to be null except during transitions from one representation level to another. Therefore, cross-model relationships do not hold at times. Moreover, ensuring that cross-model relationships hold during transitions is hard, and leads to mapping inconsistencies [—95] [—00].

A number of variants on aggregation-disaggregation exist [—96], such as full disaggregation [CALD95A], partial disaggregation [HARDY94] [BURD95], playboxes [KARR94] [COX95] [SEIDEL95] [STOBER95] and pseudo-disaggregation [CALD95B] [SMITH95] [WEAT93] [ALLEN96]. Although these variants can improve the run-time performance of multi-models that employ aggregation-disaggregation, they do not satisfy R1 or R2.

6 Concurrent Representations

Our approach for achieving effective MRM involves preserving the representations of jointly-executing models at all times. If A and B model the same phenomenon, and T^M is the sequence of times in a multi-model M constructed from A and B, then for all time-steps $[t_i, t_{i+1}]$, where $t_i, t_{i+1} \in T^M$, both A and B are executed. The representations of A and B are concurrent representations.

Maintaining concurrent representations satisfies R1. Concurrent representations permit interactions at all representation levels at all times. These interactions may change the appropriate representations at any time. Therefore, the execution of no model is restricted.

In order to satisfy R2, concurrent representations require application-specific mapping functions that translate attributes among representations. The mapping functions correspond to the relationships among attributes in the representations of jointly-executing models, i.e., the mapping functions correspond to relationships in Rel^{cross}. These functions are invoked when an interaction changes the value of some attributes. The functions translate the new values or the changes in values of attributes to new values or changes in values of related attributes.

A *Multiple Representation Entity* (MRE) is an implementation of concurrent representations. Maintaining *internal consistency*, i.e., consistency among concurrent representations, within an MRE when concurrent multi-representation interactions occur is a key challenge. For concurrent representations to be consistent with one another, changes to one representation must propagate to the other representations. We assume the presence of appropriate mapping functions to translate changes from one representation to another. Selective viewing and aggregation-disaggregation make the same assumption when emulating one level from the other or transitioning among levels. However, the assumption is not sufficient to make these approaches viable for effective MRM because the approaches continue to violate R1 and R2. We believe that our assumption is reasonable because without it the semantics of multi-models are not clear, and *no* MRM approach can be effective. Provided a designer can satisfy this assumptions, we show how to maintain internal consistency within an MRE.

6.1 Multiple Representation Entity (MRE)

An MRE preserves concurrent representations. The representation of each model in a multi-model exists within an MRE at all times. Since the representation of an entity is a subset of the representation of a model, an MRE may maintain a subset of the representations of *A* and *B* to describe one object or process present in both models. For example, in Figure 2, entity P describes an object in *A* and entities $T_{1\text{-}4}$ describe the same object in *B*. MRE E_1 consists of the representations of P and $T_{1\text{-}4}$, thus describing the object at multiple representation levels.

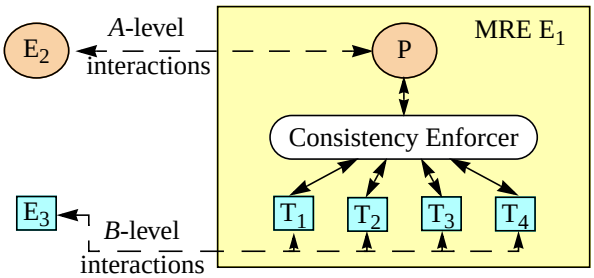

Figure 2: Multi-representation Interaction

An MRE permits interactions at all representation levels at all times. By definition, an MRE satisfies R1. An entity in either model interacts with another entity at a representation level common to both. Let E_2 be an entity in *A* and E_3 be an entity in *B* (see Figure 2). E_2 and E_1 interact at the level of *A*, which means that E_2 and P interact. Likewise, E_3 and E_1 interact at the level of *B*, which means that E_3 and $T_{1\text{-}4}$ interact. MREs disallow cross-level interactions. For example, E_2 cannot interact directly with $T_{1\text{-}4}$. Likewise, E_3 cannot interact directly with P.

The representations of jointly-executing models must be consistent at all observation times. In Figure 2, for E_1 to be internally consistent, changes to the representation of P must affect the representations of $T_{1\text{-}4}$ as well and *vice versa*. An interaction between E_2 and E_1 may result in a change to the representation of P. This change must propagate to $T_{1\text{-}4}$. Likewise, an interaction between E_3 and E_1 may result in changes to the representations of $T_{1\text{-}4}$. These changes must propagate to P.

A *Consistency Enforcer* (CE) maintains consistency among concurrent representations by propagating changes among attributes. When an interaction changes attributes in a representation in E_1, a CE changes related attributes in the other representation appropriately. Subsequently, if E_2 and E_3 view E_1 concurrently, they receive consistent views of E_1 from the representations of P and $T_{1\text{-}4}$. A CE requires mapping functions to translate changes among attributes.

An MRE is a conceptual way of designing a multi-model. An application designer constructs an MRE when designing a multi-model. Choosing the entities that constitute a single MRE is an application-specific design decision. In general, if multiple entities correspond to the same object or process, the entities should be part of the same MRE. An entity could belong to multiple MREs if an application designer considers it appropriate. The entities that constitute an MRE may reside on the same platform or across different platforms. The key requirements that the entities must satisy is that they permit interactions at all times and propagate changes to their state to other entities within the same MRE.

6.2 Mapping Functions

In order to overcome the challenge of consistency maintenance among concurrent representations, we assume the presence of mapping functions. This assumption is necessary and sufficient to maintain consistency within an MRE. We argue that this assumption is reasonable.

First, without the presence of mapping functions, the semantics of multi-models are not evident. Previous MRM approaches make a similar assumption. Selective viewing requires mapping functions to translate attributes from one representation to another. These mapping functions are invoked only once — when constructing the representation for the most detailed level. Likewise, aggregation-disaggregation requires mapping functions to translate attributes from one representation to another during aggregation and disaggregation. Mapping functions require designers to incorporate application-specific knowledge into the joint execution of multiple models.

Second, selective viewing and aggregation-disaggregation cannot guarantee effective MRM despite making a similar assumption. Since selective viewing and aggregation-disaggregation execute only one model at a time, they disallow multi-representation interactions, thus violating R1. Selective viewing satisfies R2 trivially by maintaining consistency within the representation of only

one model. Aggregation-disaggregation can violate R2 because of mapping inconsistencies among the multiple representations. In aggregation-disaggregation, when one model is executed, attributes in the representations of other models are lost. Therefore, transitioning representation levels may cause discontinuities in the values of attributes even if mapping functions exist.

6.3 Maintaining Consistency

A CE maintains internal consistency in an MRE, i.e., it ensures that an MRE exhibits temporal consistency and mapping consistency. In the interests of brevity, we present details about a CE elsewhere [—00].

Temporal Consistency: An MRE exhibits temporal consistency if the changes caused by interactions are applied consistently to all representation levels. If the multiple representations within an MRE are mutually consistent, the MRE is temporally consistent. Interactions that occur during the time-step $[t_i, t_{i+1}]$, where $t_i, t_{i+1} \in T^M$, cause changes to the state of the representations at time t_i. A CE propagates these changes in order to obtain the state at time t_{i+1}. Since an MRE is expected to be consistent only at observation times belonging to T^M, entities receive consistent views of the MRE at any observation time.

Mapping functions translate *values*, *changes in values* or *types* of attributes from one representation to another. For example, consider the T-joint in Figure 3. Model *A* may represent the T-joint with attributes such as connectedness, position and orientation. Model *B* may represent it as a pair of boards and a nail, each with attributes such as position and orientation. A mapping function must translate the positions of the boards to the position of the T-joint. Likewise, another mapping function must perform the reverse translation — from the position of the T-joint to the positions of the boards. Such mapping functions must take the values of some attributes and change them to the values of other attributes. Another pair of mapping functions must translate the orientation of the T-joint to the orientations of the boards and *vice versa*. These translations may be computationally less complex if the changes in orientations rather than the values of orientations are translated. Finally, consider the attribute of connectedness for a T-joint. Assume the system can infer that a T-joint is connected if the positions of two boards and a nail overlap[*]. A mapping function that translates the positions of the boards and nail to the connectedness of the T-joint must translate the types of the attributes as well as the values. Mapping functions must complete their translations before the time-step ends.

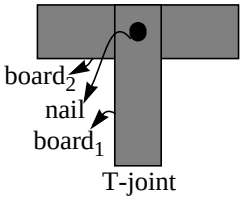

Figure 3: T-joint entity

Mapping Consistency: An MRE exhibits mapping consistency if mapping functions are reversible. Let mapping functions *f* and *g* translate an attribute *a* to another attribute *b* and *b* to *a* respectively. If $g(f(a)) = a$ and $f(g(b)) = b$, then *f* and *g* are *reversible*. An interaction initiates translations by mapping functions. Sequences of interactions initiate repeated translations. Repeated translations must not cause discontinuities in concurrent representations. Reversible mapping functions ensure that repeated translations do not cause such discontinuities.

An MRE supports the design of reversible mapping functions. For the T-joint of Figure 3, let *f* translate the board positions to the T-joint position, and *g* translate the T-joint position to the board positions. Provided no interactions occur, if *f* translates the current values of the board positions to a value for the T-joint position, then *g* translates the value of the T-joint position to new values for the board positions, the new and previous values for the board positions must be within tolerable error. If either function could have generated a number of possible values for the resultant attributes, the previous values of the resultant attributes may be taken into account in order to generate the new values. For example, if the T-joint is rotated by 180^o, invoking *f* on the values of the board positions may result in the original T-joint position. Subsequently, invoking *g* may result in board positions corresponding to no rotation, thus resulting in an intolerable change to the board positions. In contrast, if *g* took the orientation attribute or the previous values for the board positions into account, then the new positions would correspond correctly to the rotated T-joint position. Irrespective of the details, *f* and *g* must be reversible for the MRE to exhibit mapping consistency.

7 Benefits of Concurrent Representations

An MRM approach based on concurrent representations can satisfy R1 and R2. Satisfying these requirements enables eliminating problems such as chain disaggregation and network flooding, seen in previous approaches [—00]. Moreover, concurrent representations enable capturing whole-greater-than-the-sum-of-parts relationships, which occur frequently in multi-models. Finally, concurrent representations reduce the costs associated with executing multiple models jointly [—97a].

7.1 Satisfying MRM Requirements

Concurrent representations permit interactions at all representation levels at all times. In other words, concurrent representations do not require translating all interactions to the most detailed level. Consequently, they

[*] Naturally, if the boards and nail happen to lie in those positions without the boards having been nailed, the system may infer incorrectly that the T-joint is connected. Resolving this issue is out of the scope of our work, and for the purposes of this discussion, irrelevant.

do not incur the costs of executing the detailed model when unnecessary. For example, suppose a platoon model executes jointly with a model of its constituent tanks. In selective viewing, only the tank model executes. Platoon-level interactions must be translated to possibly many tank-level interactions, each possibly changing the representations of the corresponding tanks. In an MRE, platoon-level interactions change the representation of the platoon. Changes to the platoon representation propagate to the tank representations. In this respect, concurrent representations capture the main benefit of aggregation-disaggregation — reducing complexity when possible.

Concurrent representations always maintain attributes at all representation levels. In contrast, in aggregation-disaggregation, attributes are discarded after a translation. In an MRE, attributes at *all* levels are retained after a translation. Consequently, mapping functions can utilise previous values of attributes in order to generate new values, thus avoiding inconsistencies.

7.2 Capturing Whole-Greater-than-the-Sum-of-Parts Relationships

Aggregate and disaggregate entities may be whole and parts of one another. Whole-and-parts relationships occur frequently in multi-models. For example, in battlefield simulations a number of tanks may be considered as parts of a platoon, or a number of regiments may be considered as parts of a division. Likewise, in multi-resolution graphics, a number of triangles may be considered as parts of an entire surface, and in molecular models, a number of atoms may be considered as parts of a molecule.

A valid concern when aggregate and disaggregate models execute jointly is that the values of some aggregate attributes may be greater than the sum of the values of corresponding disaggregate attributes, i.e., the whole is greater than the sum of its parts. This concern has been called emergent behaviour problem [WIM86] or the configuration problem [HORR92]. For example, tanks may fight with greater strength when configured as a platoon. This increase in strength may be attributable to the presence of a commander who coordinates and guides activities (as is common in the case of military units) or factors like morale, fatigue and national resolve, which may not fit into the tank model but play a part in the performance of the platoon model. As another example, weak forces in atomic models may be ignored since their effect on the position of atoms may be negligible. However, in molecular models, these forces together may influence the positions of atoms significantly. The precise relationships between the platoon's strength and the tanks' strength and the atomic forces and the molecular forces must be captured by mapping functions that translate attributes among representations.

Selective viewing does not capture whole-greater-than-the-sum-of-parts relationships. Since only the model for the parts is executed, whole-greater-than-the-sum-of-parts relationships can be captured only if information outside the attributes of each part is present. Typically, an entity maintains attributes relevant only to its own execution. Therefore, the behaviour of an entity when it executes as part of a whole is not distinct from its behaviour when it executes individually. Consequently, information not present in the entity must be used to distinguish these behaviours. Maintaining such information is tantamount to executing multiple models.

Aggregation-disaggregation captures whole-greater-than-the-sum-of-parts relationships, but introduces mapping inconsistency because information is lost during transitions. For example, tanks in a platoon may have manœuvred into a favorable position, thus causing the strength of the platoon to be greater than the sum of the strengths of the tanks. At this point, transitioning to the platoon model and back to the tank model may cause the tanks to be placed in doctrinal formation (since the tanks' previous positions are lost). This placement may result in a platoon strength that is the sum of the strengths of the tanks, thus reducing the strength of the platoon. Worse, the disaggregation process may be carried out by a different system than the one executing the aggregate model. When more than one of these systems is active in a distributed simulation at the same time, the disaggregation performed by each of them may generate different results.

Concurrent representations aid in the construction of mapping functions that capture whole-greater-than-the-sum-of-parts. Since in concurrent representations attributes at all levels are always present, mapping functions that avoid inconsistency can be designed. It is the responsibility of the designer to identify and encode such relationships within mapping functions.

7.3 Reducing Costs

Concurrent representations reduce the costs associated with executing multiple models jointly. There are two major costs associated with joint execution: simulation cost and consistency cost. Simulation cost is the cost of simulating the entities in the multi-model. Consistency cost is the cost of maintaining consistency among the models. Since selective viewing executes only the most detailed model, it results in low consistency costs, but high simulation costs. In contrast, aggregation-disaggregation results in low simulation costs, but high consistency costs because it transitions among models. An approach based on concurrent representations may balance these two costs, thus resulting in lower total costs than either of the previous approaches. We present a comparison of the costs associated with MREs and other MRM approaches elsewhere [—97A].

8 Conclusions

An approach for Multi-Representation Modelling (MRM) is effective if it permits the consistent joint execution of multiple models of the same phenomenon. An effective approach must permit the execution of any and all of the models at all times, and maintain consistency among them. In other words, the approach must satisfy the requirements of multi-representation interaction (R1) and multi-representation consistency (R2).

Previous MRM approaches, such as selective viewing and aggregation-disaggregation have been unable to satisfy R1 and R2. Although these approaches may be useful for particular multi-models despite not satisfying these requirements, they fail for other multi-models. The problems encountered by these approaches can be eliminated by an effective MRM approach.

Concurrent representations can lead to an effective approach for MRM. Maintaining concurrent representations means preserving the representations of jointly executing models at all times and permitting interactions to change them at any time, possibly concurrently. A Multiple Representation Entity (MRE) is a technique for maintaining concurrent representations. A key challenge with an MRE is maintaining consistency among its representations in the presence of concurrent interactions. We assume the existence of appropriate mapping functions for translating attributes from one representation to another. This assumption does not make MRM trivial, because previous approaches continue to exhibit problems even if they make a similar assumption. We expect an approach based on concurrent representations to resolve the problem of making multiple models execute jointly and consistently.

9 References

ABADI95 Abadi, M., Lamport, L., *Conjoining Specifications*, ACM Transactions on Programming Languages and Systems, Vol. 17, No. 3, May 1995.

ALHIR98 Alhir, S. S., *UML in a Nutshell*, O'Reilly & Associates Inc., ISBN 1-56592-448-7, 1998.

ALLEN96 Allen, P. M., Valle, A. N., *An approach to managing dissimilar unit interactions in constructive/virtual simulation linkage*, 14th DIS Workshop on Standards for the Interoperability of Distributed Simulations, September 1996.

BURD95 Burdick, C. D., *Interoperability of Simulations with Different Levels of Resolution*, Defense Modeling and Simulation Office Workshop, November 1995.

CALD95A Calder, R. B., Peacock, J. C., Panagos, J., Johnson, T. E., *Integration of Constructive, Virtual, Live, and Engineering Simulations in the JPSD CLCGF*, 5th Conference on Computer Generated Forces & Behavioral Representation, May 1995.

CALD95B Calder, R. B., Peacock, J. C., Wise, B. P. Jr., Stanzione, T., Chamberlain, F., Panagos, J., *Implementation of a Dynamic Aggregation/ Deaggregation Process in the JPSD CLCGF*, 5th Conference on Computer Generated Forces & Behavioral Representation, May 1995.

COX95 Cox, A., Maybury, J., Weeden, N., *Aggregation Disaggregation Research — A UK Approach*, 13th DIS Workshop on Standards for the Interoperability of Distributed Simulations, September 1995.

DAVIS92 Davis, P. K., *An Introduction to Variable-Resolution Modeling and Cross-Resolution Model Connection*, Conference on Variable-Resolution Modeling, May 1992.

DAVIS93 Davis, P. K., Hillestad, R. J., *Families of Models that Cross Levels of Resolution: Issues for Design, Calibration and Management*, Winter Simulation Conference, 1993.

HARDY94 Hardy, D., Healy, M., *Constructive & Virtual Interoperation: A Technical Challenge*, 4th Conference on Computer Generated Forces & Behavioral Representation, May 1994.

HOP79 Hopcroft, J. E., Ullman, J. D., *Introduction to Automata Theory, Languages, and Computation*, Addison Wesley Publishing Company Inc., ISBN 0-201-02988-X, 1979.

HORR92 Horrigan, T. J., *The "Configuration Problem" and Challenges for Aggregation*, Conference on Variable-Resolution Modeling, May 1992.

KARR94 Karr, C. R., Root, E., *Integrating Aggregate and Vehicle Level Simulations*, 4th Conference on Computer Generated Forces & Behavioral Representation, 1994.

LAM94 Lamport, L., *The Temporal Logic of Actions*, ACM Transactions on Programming Languages and Systems, Vol. 16, No. 3, May 1994.

—95 …

NAT95 Natrajan, A., Nguyen-Tuong, A., *To disaggregate or not to disaggregate, that is not the question*, ELECSIM, April-June, 1995, Technical Report CS-95-18, Department of Computer Science, University of Virginia, 1995.

—96 …

NAT96 Natrajan, A., Reynolds Jr., P. F., Srinivasan, S., *Consistency Maintenance using `UNIFY`*, Technical Report CS-95-28, Department of Computer Science, University of Virginia, 1996.

—97A …

NAT97 Natrajan, A., Reynolds Jr., P. F., Srinivasan, S., *A Flexible Approach to Multi-Resolution Modeling*, Parallel and Distributed Simulation, June 1997.

—00 …

NAT00 Natrajan, A., *Consistency Maintenance in Concurrent Representations*, Ph.D. Dissertation, Department of Computer Science, University of Virginia, 2000.

OMT98 U.S. Department of Defense, *High Level Architecture Object Model Template Specification Version 1.3*, IEEE P1516.2, Standard for Modeling and Simulation, April 1998.

—97B …

REYN97 Reynolds Jr., P. F., Natrajan, A., Srinivasan, S., *Consistency Maintenance in Multi-Resolution Simulations*, ACM Transactions on Modeling and Computer Simulation, Vol. 7, No. 3, July 1997.

RUM91 Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen, W., *Object-Oriented Modeling and Design*, Prentice Hall PTR, ISBN 0-13-629841-9, 1991.

SEIDEL95 Seidel, D. W., King, B. C., Burke, C. D., *AIM Approach to Simulation Interoperability*, The MITRE Corporation, Preliminary Draft, July 1995.

SHLAER92 Shlaer, S., Mellor, S. J., *Object Lifecycles: Modeling the World in States*, Prentice Hall PTR, ISBN 0-13-629940-7, 1992.

SMITH95 Smith, R. D., *The Conflict Between Heterogeneous Simulation and Interoperability*, 17th Inter-Service/ Industry Training, Simulation, and Education Conference (I/ITSEC) Proceedings, November 1995.

STOBER95 Stober, D. R., Kraus, M. K., Foss, W. F., Franceschini, R. W., Petty, M. D., *Survey of Constructive+Virtual Linkages*, 5th Conference on Computer Generated Forces & Behavioral Representation, May 1995.

TURING50 Turing, A. M., *Computing Machinery and Intelligence*, Mind, Vol. 59, October 1950.

WEAT93 Weatherly, R. M., Wilson, A. L., Griffin, S. P., *ALSP — Theory, Experience and Future Directions*, Winter Simulation Conference, 1993.

WIM86 Wimsatt, W. C., *Heuristics and the Study of Human Behavior*, in Fiske, D., Shweder, R., eds., Meta-Theory in the Social Sciences: Pluralisms and Subjectives, University of Chicago Press, 1986.