Consistency Maintenance in Multi-Resolution Simulations

Paul F. Reynolds, Jr.
Dept. of Computer Science, Thornton Hall, University of Virginia, Charlottesville, VA 22901, USA.
Tel: (804) 924-1039. Email: *reynolds@virginia.edu*, URL: *http://www.cs.virginia.edu/~pfr/*

Sudhir Srinivasan
Mystech Associates Inc., 5205 Leesburg Pike, Suite 1200, Falls Church, VA 22041, USA.
Tel: (703) 671-8680. Email: *sudhirs@mystech.com*, URL: *http://www.mystech.com/~sudhirs/*

Anand Natrajan
Dept. of Computer Science, Thornton Hall, University of Virginia, Charlottesville, VA 22901, USA.
Tel: (804) 982-2291. Email: *anand@virginia.edu*, URL: *http://www.cs.virginia.edu/~an4m/*

Sponsor: Defense Modeling & Simulation Office, 1901 N. Beauregard St., Suite 504, Alexandria, VA 22311, USA. Tel: (703) 998-0660

Abstract

Simulations that run at multiple levels of resolution often encounter consistency problems because of insufficient correlation between the attributes at multiple levels of the same entity. Inconsistency may occur despite the existence of valid models at each resolution level. Cross-Resolution Modeling (CRM) attempts to build effective multi-resolution simulations. The traditional approach to CRM — aggregation-disaggregation — causes chain disaggregation and puts an unacceptable burden on resources. We present four fundamental observations that would help guide future approaches to CRM. These observations form the basis of an approach we propose that involves the design of Multiple Resolution Entities (MREs). MREs are the foundation of a design that incorporates maintaining internal consistency. We also propose maintenance of core attributes as an approach to maintaining internal consistency within an MRE.

Subject Categories: I.6.5 [**Model Development**], Modeling Methodologies
General Terms: Designs
Keywords: multi-resolution modeling, multi-resolution simulations, multiple resolution entity, consistency maintenance

1 Introduction

Cross-Resolution Modeling (CRM) [Davis93] is concerned with resolving conceptual and representational differences that arise from multiple levels of resolution in simulations that are joined for a common objective. CRM poses a very significant challenge, particularly in cases where the simulations were designed and implemented independently. The crux of the problem can be appreciated by considering what is required to accurately simulate an object *and* its constituents concurrently. For example, the abstraction *convoy* may have attributes such as position, velocity, orientation and state of repair. At a more concrete level, the convoy can be viewed as trucks that have attributes such as position, velocity, orientation, state of repair, fuel level, gross weight, carrying capacity, number and location of occupants, etc. If the convoy abstraction and its constituent trucks are modeled concurrently, all interactions with the convoy abstraction and its constituents in overlapping periods of time must be accurately reflected at both levels. We address the CRM problem here. We do not solve it, but we do provide some fundamental observations we believe are essential parts of any solution that may exist, and we introduce the notion of a *Multi-Resolution Entity* (MRE) as the basis for addressing the CRM problem in a consistent manner.

We recognize the importance of model validity and encourage its pursuit. However, our work does not focus on model validity. Even if we assume that the models to be linked are valid, inconsistency can arise in the linkage. Consistency issues arise when *Low Resolution Entities* (LREs), for example, corps, interact with *High Resolution Entities* (HREs), for example, tanks. A common solution approach is to dynamically change the resolution of an LRE (or HRE) to match the resolution of other encountered entities. This dynamic change is called *aggregation* (HREs $\rightarrow$ LRE) or *disaggregation* (LRE $\rightarrow$ HREs). The problem of linking simulations at different levels of resolution has thus come to be known as the *aggregation-disaggregation problem*. We prefer Davis's term *Cross-Resolution Modeling*. A disaggregated LRE that re-aggregates itself to interact with another LRE and then later disaggregates, may put itself in a state that it could not have otherwise achieved over the same period of time. Also, this dynamic aggregation-disaggregation approach incurs problems such as chain disaggregation, network flooding, transition latency and mapping problems between levels, all of which we address in this paper. We have found that existing solutions meant to solve some or all of these problems leave the central consistency problem unresolved. Consistency issues arise mainly due to failure to appreciate the fundamental observations we present here, and the consequent *ad hoc* solutions. A more unified approach, such as that offered by MREs, is required.

The bulk of the work that has been done on CRM is related to military simulations, primarily training simulations. A common scenario is the coupling of a simulation that models military forces at an abstract level, say platoons, battalions or corps, with a simulation that models individual battlefield entities, such as tanks or wheeled vehicles. The CRM problem arises if simulation A models a given platoon that is also modeled as its constituent tanks in simulation B. A form of this problem has occurred in the majority of the couplings of military simulations that have been attempted over the past ten years. The result has been the evolution of a number of point-to-point solutions, the sum of which still does not comprise a satisfactory solution to the CRM problem. Since the bulk of the attempts have been associated with the military, our presentation uses military examples. However, the technology we explore applies equally well to problems such as socio-economic, biological, or environmental modeling.

The fundamental observations we present here are exactly that, *observations*. While they are presented in a less-than-rigorous manner, we nonetheless present strong arguments for their existence. Our observations are fundamental because any general solution to the CRM problem *must* take them into account. They address the general ineffectiveness of linking separate entities represented at different levels of abstraction, the necessity of maintaining consistency in multi-level representations of the same abstraction, the need to address the dependence that exists among the multiple levels of resolution of a single abstraction, and the need to address temporal consistency. Altogether, these observations lead to a set of requirements for satisfying the CRM problem, albeit, an incomplete set because they are not sufficient to completely resolve the CRM problem. However they narrow it to the core problem of how to maintain consistency in the multiple levels of resolution of a single abstraction.

If there is a way to maintain consistency within multiple, concurrent representation levels of an abstraction, MREs are our recommended approach for capturing it accurately. MREs are offered in part in response to the largely *ad hoc* approaches that have been pursued in previous attempts to address the CRM problem. MREs make possible the isolation of the issues related to maintaining consistency, and they enable possible efficiencies through concepts such as *core attributes*. MREs offer to CRM what modularity offers to software development: a common semantic framework and the attendant opportunities for efficiency enhancement.

Unfortunately, all previous attempts to maintaining consistency in multi-resolution simulations have made coarse assumptions about consistency. The fundamental observations and consistency maintenance approach (MREs) we describe here are meant to support consistency requirements in multi-resolution simulations. In the remainder of this paper we present definitions, explore CRM issues and the problems that have arisen in current solutions, present and argue the necessity of four fundamental observations on CRM and present our approach to enabling consistency maintenance in multi-resolution simulations: MREs.

2 Definitions

We present definitions for terms we use. Some of these definitions are based on those in [AMG95].

- ***Object***: A fundamental conceptual representation that reflects the real world at levels of abstraction appropriate for a planned simulation.
- ***Entity***: A representation of an object in a simulation.
- ***Model***: A mathematical abstraction of the behavior of an object, usually instantiated in simulation source code.
- ***Resolution***: The level of abstraction at which an entity is modeled.
- ***Simulation***: A dynamic representation of one or more objects, involving some combination of executing code, control/display interface hardware and interfaces to real-world equipment.
- ***Multi-level Simulation***: A simulation that involves entities at different levels of resolution.
- ***High Resolution Entity (HRE):*** An entity at a low level of abstraction, typically modeling a single object.
- ***Low Resolution Entity (LRE)***[1]: An entity at a high level of abstraction, typically modeling several objects.
- ***Multiple Resolution Entity (MRE)***: A *conceptual* entity that can interact at multiple levels of resolution concurrently by maintaining consistency among corresponding attributes at different levels of resolution. The

MRE is a concept and as such, may not correspond to a single physical piece of software. In practice, the MRE will likely be comprised of several pieces of software, either newly-developed or legacy simulations, acting together to yield a consistent multi-resolution simulation.

- ***Aggregation***: Modeling of a collection of HREs as a single LRE.
- ***Disaggregation***: Decomposition of an LRE into its constituent HREs.
- ***Effective Linkage***: A linkage between entities, possibly at different levels of resolution, that meets certain effectiveness criteria prescribed for that linkage. For example, if the entities are two training simulations, the effectiveness criteria could be fidelity and validity sufficient for the training function; for engineering simulations, it could be accuracy in output metrics to a specific, quantifiable range.
- ***Ghosting***: The act of reflecting the attributes of an entity being modeled by another simulation.
- ***VV&A***: Verification, Validation and Accreditation — a process that determines whether a simulation (or linkage) meets its prescribed effectiveness criteria.

Note: Levels of resolution and levels of aggregation (or abstraction) are inversely related: high-resolution means low level of aggregation, and low-resolution means high level of aggregation.

3 Cross-Resolution Modeling (CRM)

Cross-resolution modeling is applicable when simulations at different levels of resolution are required to interoperate. Crucial to CRM are assumptions made about the simulation's levels of resolution. Often, two simulations that are required to work together have different characteristics that make interoperation difficult. One simulation may be at a higher resolution because it models entities that are very fine-grained, whereas the other may be at a lower resolution because its entities are coarse-grained. Assumptions about objects, events, interactions and environment may be different. The fundamental processes in the simulation, such as line-of-sight and dead-reckoning, may have different algorithms because of the difference in resolution. The simulations may manage time differently: discrete-event versus time-stepped versus continuous. Also, the time-steps at which the simulations proceed may be vastly different. Ensuring that such simulations interact with each other meaningfully is the heart of cross-resolution modeling.

Variable Resolution Modeling (VRM), proposed by Davis [Davis93], is related to CRM in that it deals with designing simulations that operate at different levels of resolution. Davis's technique for VRM is based on process hierarchies where processes lower in the hierarchy typically are at a higher level of resolution. Designing with these hierarchies in mind facilitates the construction of accurate models that can operate at any desired level of resolution.

The common approach to CRM is aggregation-disaggregation. Aggregation-disaggregation ensures that entities interact with each other at the same level by forcing one entity to go to the level of the other. Typically, if a low-resolution entity (LRE) interacts with a high-resolution entity (HRE), the LRE is decomposed into its constituents in a process known as disaggregation. LRE-LRE interactions would be at the LRE level. A disaggregated LRE may be re-aggregated so that it can interact subsequently at the LRE level. In following sections, we discuss aggregation-disaggregation issues that must be addressed.

3.1 Temporal Inconsistency

Temporal inconsistency exists when two entities have conflicting or inconsistent representations of the state of a third entity at overlapping simulation times. This problem is observed commonly in linkages that proceed at vastly disparate time-steps at different resolution levels. Suppose entities E_1 and E_2 interact once every minute and E_1 and E_3 interact once every millisecond. E_1 essentially commits to maintaining the state communicated to E_2. However, while E_2 processes its last interaction with E_1, E_3 may cause E_1 to change state many times over, thus violating E_1's commitment to E_2. This causes a temporal inconsistency because E_2 and E_3 have inconsistent representations of E_1 at the end of the larger time-step. Temporal inconsistency directly degrades the effectiveness of the linkage.

3.2 Mapping Inconsistency

Mapping inconsistency occurs when an entity undergoes a sequence of transitions across levels of resolution resulting in a state it could not have achieved in the simulated time spanned by that sequence. Any scheme in which

1. Classification of an entity as an HRE or LRE depends on its resolution level relative to other relevant entities. Thus, a tank classified as an HRE in a training simulation may be classified as an LRE in an engineering simulation.

entities transition across resolution levels (e.g., aggregation-disaggregation) must consistently map attributes across levels. Specifically, the translation should enable switching levels without a change in the attributes, provided no other interactions occur. Poor translation strategies cause “jumps” in the state of entities. A jump in visual perception is caused when the perceived changed position of an entity violates simulation semantics due to rapid translations between states. The aggregate-level information may be insufficient in providing disaggregate-level consistency. In other words, a disaggregated-to-aggregated transition may lose some information pertaining to the HREs, say position. Consequently, a second transition, this time aggregated-to-disaggregated, may result in a disaggregated state inconsistent with the first disaggregated state because a standard algorithm or doctrine has been applied to position the entities [Clark94, France93, Davis93]. While perfectly state-maintaining translation strategies are desirable, often these may not be found readily. In such cases, the potential perceptual inconsistencies arising due to translations from one state to the other must be addressed differently.

3.3 Chain Disaggregation

Chain disaggregation is best explained by an example. Suppose an HRE H begins interacting with an LRE L. Typically, L is disaggregated so that L and H can interact at the disaggregate level. However, other entities that may have been interacting with L would now also have to disaggregate. It is easy to extend this reaction to yet other entities that are forced to disaggregate because an LRE they were interacting with disaggregated. Figure 1 illustrates the problem by showing an HRE (shaded) coming into contact with an LRE (unshaded). Subsequently, all LREs have

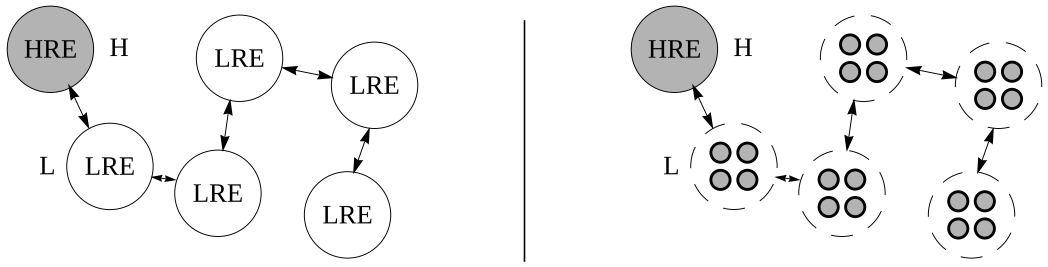

FIGURE 1: Chain Disaggregation

to disaggregate in order to be able to interact at the same level. The domino effect caused by the initial disaggregation is called chain disaggregation, also known as spreading disaggregation in the literature. Chain disaggregation causes the number of simulated entities to increase rapidly. This increases the load on processors and the network.

3.4 Transition Latency

During disaggregation, a translation must be made from the state of the aggregate unit to the state of its disaggregate constituents. This involves a set-up time, time to populate disaggregate attributes from the aggregate and initiation of protocols to place entities. A similar protocol may exist when going from the disaggregate to the aggregate level. The time taken to effect an aggregation or disaggregation is known as the transition latency. Transition latency can be significantly high depending on the complexity of the protocol. For example, a protocol in [Robkin92] takes on the order of 10 seconds to complete the aggregation process. High transition latencies are incompatible with real-time constraints, for example, in human-in-the-loop simulations, because they may cause perceptual or conceptual inconsistencies. An entity that does not change position during a transition period, and then suddenly undergoes a large displacement at the end of the transition period causes a perceptual inconsistency or “jump”. A conceptual inconsistency may be caused when it takes so long for an entity to disaggregate in order to comply with a request made by another entity that the request becomes obsolete.

3.5 Thrashing

When an entity undergoes rapid and repeated transitions from one level to the other, it thrashes. For example, an LRE L may disaggregate on commencing interactions with some HRE H. When H moves out of range, L may revert to the aggregate level. However, H’s trajectory may cause L to repeatedly change levels within a short time. This would cause L to “flip-flop” levels, each time incurring the overhead associated with making a level change. While thrashing depends primarily on the triggering policy that causes a change of level, it is nevertheless an issue that must be addressed in the design of multi-level simulations itself. High transition latencies compounds the problems due to thrashing because it causes some entities to spend considerable amounts of time just changing levels.

3.6 Network Flooding

The network is projected to be a bottleneck in distributed simulations. Network resources may be strained by the acts of aggregation and disaggregation, depending on the scheme used. Disaggregation creates new entities, each of which could be a sender and/or receiver of messages. Clearly, even if only the entity state messages generated by all the entities are taken into account, this is an increase in network traffic. Also, aggregation and disaggregation protocols typically require the exchange of many control messages — an overhead that must be incurred every time a change of level occurs. This can reduce the effective throughput of the network. Frequent changes of level and the large number of entities may put an unacceptable burden on the network.

4 Previous Work

Several projects have undertaken the task of constructing a cross-resolution linkage between so-called legacy simulations — simulations previously designed to operate independently. Since the DoD has been the prime mover in the distributed simulation area, it is not surprising that most of these projects are in the military domain (as is evident from the examples that follow). The state-of-the-art in CRM can be described as being moderately successful in that linkages have been effective to some extent in each project but a general technique and associated theory are lacking — for the most part, solutions have been point-to-point. We describe the general techniques that have been used in these projects and indicate their limitations.

4.1 Full Disaggregation

As the name suggests, full disaggregation involves disaggregating the entire LRE into its pre-designed constituent HREs. Typically, this happens when the LRE establishes contact (sensor, line-of-sight, etc.) with an HRE. The limitations of a complete disaggregation are obvious: in most cases, it is superfluous (consider a unit of 100 entities disaggregating completely due to contact with a single HRE); due to chain disaggregation, it can easily lead to disaggregation of all entities in the simulation; and it has the potential to place an unacceptable burden on system resources by increasing the number of entities in the simulation. These limitations restrict the applicability of full disaggregation to small-scale situations, for example, those involving few tens of entities [Calder95].

4.2 Partial Disaggregation

Partial disaggregation attempts to overcome the main limitations of full disaggregation by disaggregating only a part of an LRE rather than the entire LRE. As seen in Figure 2, a partition is created inside LRE L_2 such that only a

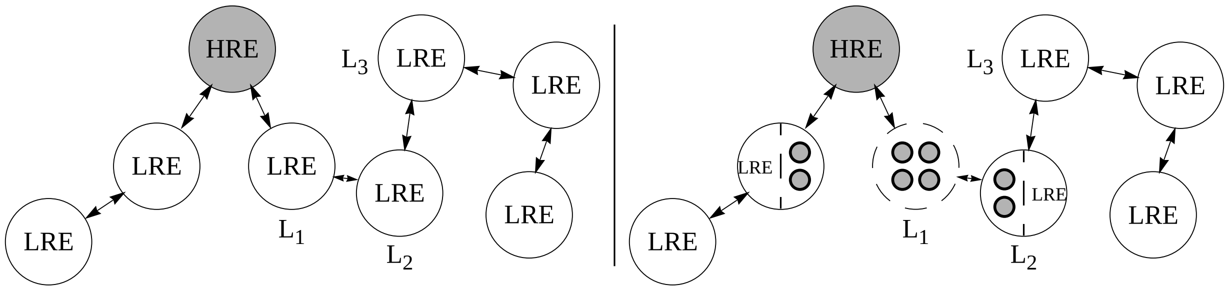

FIGURE 2: Partial Disaggregation

part of L_2 is disaggregated into HREs that interact with the disaggregated constituents of LRE L_1; the remaining part of L_2 is left as an LRE to interact with LRE L_3. This approach is taken in the `BBS/SIMNET` (Brigade-Battalion Battle Simulation/Simulation Networking) linkage [Hardy94, Burd95] in which a `BBS` entity that engages a `SIMNET` entity is partitioned such that one part disaggregates and fights a disaggregate-level battle in the `SIMNET` world, while the other part remains aggregated and fights aggregate-level battles in the `BBS` world.

Partial disaggregation is the most general technique proposed thus far. It is often seen as a solution to the chain disaggregation problem. As seen in Figure 2, partial disaggregation clearly has the potential to control chain disaggregation. However, this potential depends on the feasibility of constructing a partition inside an LRE. The criteria for constructing the partition must be decided carefully to prevent partial disaggregation from degenerating into full disaggregation. The common approach uses a pre-determined region of the simulated domain, called a *playbox*, to partition the LREs. Conceptually, the playbox may be defined in any domain, but for exposition, we assume a geographic domain such as a simulated battlefield. A portion of the domain is demarcated as a playbox. Entities inside the playbox are disaggregated while those outside remain at the aggregate level. An LRE that crosses into the playbox must be disaggregated; likewise, when all the disaggregated constituent entities of an LRE leave the playbox, they are aggregated into the LRE. The playbox is typically static, although it can be dynamic.

Playboxes may force entities to disaggregate unnecessarily (i.e., when an entity enters the playbox but does not interact with others). Further, thrashing can occur very easily when the trajectory of an entity causes it to move in and out of the playbox rapidly. Finally, adequately general solutions do not exist to address interactions across the boundary of the playbox. Static playboxes have the additional limitation that they artificially constrain the region in which LREs and HREs may interact meaningfully. The `AIM` (`AWSIM` Interoperability with `ModSAF`) project links `AWSIM` (Air Warfare Simulator), an aggregate-level aircraft simulator with `ModSAF` (Modular Semi-Automated Forces), a disaggregate-level simulator capable of simulating tanks and aircraft [Seidel95]. `AIM` uses a playbox approach wherein the `ModSAF` region of interest defines the playbox. When `AWSIM` aircraft enter the playbox, they are disaggregated from `AWSIM` and the responsibility of modeling them is relinquished to `ModSAF`. Other projects that use playboxes are `Eagle/BDS-D` [Stober95] and `Abacus/ModSAF` [Cox95].

Another approach is to partition the LRE based on some application-specific criteria such as command-and-control decisions. Even so, the criteria may be inadequate for creating partitions in all cases, leading to full disaggregation. We argue that the fundamental observations presented in Section 6 must be taken into account in designing a general partitioning criterion (and hence a scalable cross-resolution linkage technique).

4.3 Pseudo-Disaggregation

Consider a situation where an HRE requires the attributes of the constituent HREs of some LRE but does *not* interact with them. A common example is an Unmanned Airborne Vehicle (UAV) used to obtain aerial pictures of the ground situation which are processed for details of observed entities. Since the LRE is an abstraction for convenience of simulation, any LREs in the picture obtained by a UAV simulator may need to be decomposed into their constituent HREs. In this case, disaggregating the LREs is wasteful since only a perception of the constituent HREs is required, and there is no interaction with them. In pseudo-disaggregation (Figure 3), the HRE receives low-resolution

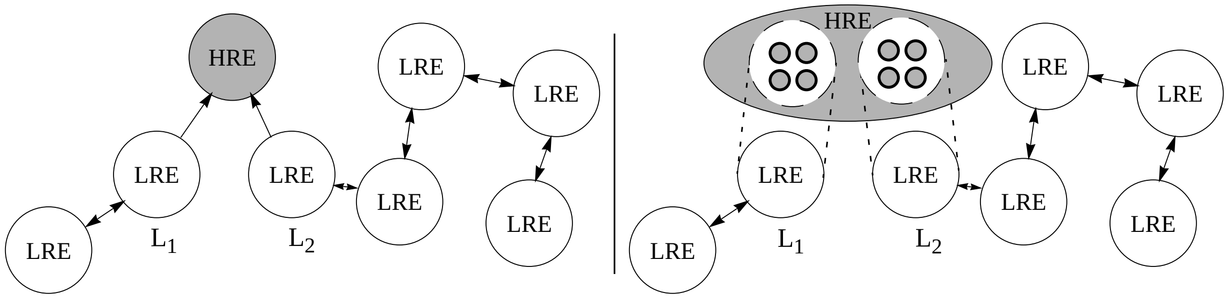

FIGURE 3: Pseudo-disaggregation

information from the LRE and *internally* disaggregates the information to obtain the high-resolution information it needs. Clearly, this technique is applicable only when the interaction is unidirectional — i.e., it does not require any behavioral modeling of the constituent HREs of L_1 and L_2. Further, the algorithms used by the HRE to locally disaggregate L_1 and L_2 must be the same as the ones L_1 and L_2 would use to disaggregate themselves, if required. This limits the scalability of pseudo-disaggregation since each HRE may be required to know how to disaggregate every LRE in the simulation. Pseudo-disaggregation is employed by `JPSD` (Joint Precision Strike Demonstration) [Calder95], `TACSIM/CBS` (Tactical Simulator/Corps Battle Simulator) [Smith95] and `Eagle/BDS-D` [Stober95].

4.4 Cross-Level Interactions

Aggregation and disaggregation are techniques that facilitate interactions at the same level of resolution. However, despite these techniques, most projects encounter interactions spanning levels of resolution. For example, in an *indirect fire* situation, two entities could engage in combat *without* direct interaction (as in long-range artillery fire). Due to the indirect nature of the engagement, disaggregation is not triggered[2], and the interaction spans the two levels of resolution. We refer to such interactions as cross-level interactions.

One aspect of cross-level interactions is meaningfully translating the semantics from one level to another. Vastly different assumptions, models, algorithms and purposes make this a very difficult task. A second aspect is reconciling differences in time-steps at the different levels of resolution. Thus, the fundamental issues of a cross-resolution

2. Note, forcing a disaggregation could lead to chain disaggregation, and is therefore undesirable.

linkage remain despite the techniques of aggregation and disaggregation. In a sense, cross-level interactions are a microcosm of the larger problem of cross-resolution modeling.

Many projects have claimed success in modeling cross-level interactions. Typically, point-to-point solutions using approximations such as ignoring time-step differentials, smoothing and clustering have been used and the results reported to be sufficiently realistic without due VV&A process. We argue that such techniques cannot provide effective linkages (Section 6.1). The findings (to date) of the `AIM` project [Seidel95] support this claim in that designers could not determine important model parameters such as hit-kill probabilities for cross-level interactions.

5 Effectiveness

Before making observations on the general nature of multi-level simulations, we need to acquire a notion of effectiveness. Criteria for effectiveness of a simulation should be specified within the requirements of the simulation. Whether a simulation meets these requirements or not should be determined by an appropriate VV&A procedure. For training simulations, effectiveness may be indicated by the perception of the training experts with regard to how well the simulation reflected reality in significant respects. A term often used in the training community is *fair fight*, which signifies an engagement in which neither party is able to deduce and utilize information about the training system (that they would not be able to deduce in a real situation) to gain an unfair advantage. For example, due to an artifact of the simulation, an aircraft may continue to be perceived for some time after having been destroyed. This artifact could be employed to draw additional fire and thus force consumption of ammunition without sustaining losses. Similarly, there have been reports of crews in tank simulators being able to identify other tanks as being controlled by humans rather than by computer-generated forces by tracking their movements.

The fair-fight concept is relevant to simulations since they approximate reality and there is potential to exploit knowledge of these approximations to gain advantage. This is especially true of cross-resolution linkages where a basic theory is still developing and design choices have been made arbitrarily for the most part. An important consideration in designing effective simulations and linkages is ensuring that there do not exist events or interactions that are artifacts of the simulation or the linkage. Determining whether a training simulation is effective is akin to performing the Turing test [Turing50]: if the requirements designers establish that the training audience is unable to differentiate between simulation and reality and if the lessons learned are valid, the simulation is effective.

It is important to understand the difference between an unfair fight and what military analysts call the “fog of war”. The fog of war refers to circumstances — typically large numbers of concurrent events — that make it difficult to maintain a coherent picture of the battle, leading to unexpected events. Unfair fights, on the other hand, result from shortcomings in the design of the training system and have no counterparts in a real battle. Often, inconsistencies due to the training system are incorrectly passed off as being a part of the fog of war. While creating simulations that pass the Turing test completely is difficult, an important goal of simulationists should be to reduce the discrepancies that cause a simulation to fail the test [Petty94].

The `AIM` project [Seidel95] lists in detail, the anomalies resulting from the linkage. We believe situations can be created wherein each of the projects surveyed in Section 4 will fail the effectiveness test, due to a variety of reasons: temporal inconsistency, time-step differential, mapping inconsistency and arbitrary approximation. We discuss guidelines which address these issues next.

6 Fundamental Observations

We present some fundamental observations regarding the problem of cross-resolution modeling. We refer to these as *observations* rather than theorems because the truth of their statements is argued informally rather than proven rigorously. We argue these observations are fundamental because any general solution to the cross-resolution problem *must* take them into account. These observations are a result of a thorough analysis of the issues concerning cross-resolution modeling and the projects surveyed in Section 4. It is our belief that these fundamental observations will provide the foundation for the theory of cross-resolution modeling and guide the development of long-term solutions to the various issues in cross-resolution modeling.

6.1 Fundamental Observation 1: Appropriate Levels of Resolution

FO-1: With few exceptions, effective linkage requires entities to be modeled at appropriate levels of resolution.

“Appropriate levels of resolution” may be defined as the levels at which semantics are compatible. Consider a linkage between two models with entities, E_A and E_B at two different levels of resolution (L_A and L_B respectively), as shown in Figure 4. Essentially, FO-1 states that for most applications, in order to interact with each other, either E_A must be represented at L_B or E_B must be represented at L_A. In other words, only those linkages that follow a combination of a vertical and a horizontal link can be effective — a diagonal linkage cannot.

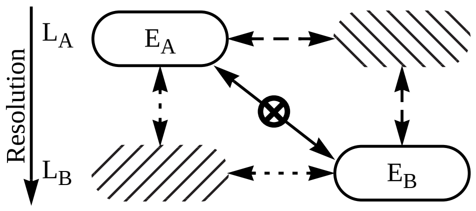

FIGURE 4: Fundamental Observation 1

To see why this observation is true, consider a military training simulation where the dimension of resolution is the size of a fighting unit (e.g., platoon vs. single tank). Here, E_A may be a division of tanks being modeled in a low-resolution simulation such as `CBS` while E_B may be a single, self-contained (manned) tank simulator. On the one hand, engagements in the constructive simulation are simulated typically by solving equations that take into account the relative strengths of the engaging parties — actual firing of weapons and destruction of individual tanks are not simulated. On the other hand, engagements involving the individual tank simulator are simulated entirely on the basis of actions taken by the parties involved in the engagement (for example, the human crew of the tank). These involve simulation of detailed actions such as sighting, target acquisition, firing, detonation and damage assessment. It is clear that engagements are handled in entirely different ways at the two levels of resolution. In general, models at different levels of resolution are designed for different purposes and consequently, have different foci. What is relevant at one level may not be relevant at another (and therefore may not be modeled there). The crew members inside an individual tank simulator expect to see individual targets through their sensors. Presenting them with an aggregated view of a tank division will be ineffective (if the effectiveness criterion is the visual fidelity of the engagement).

Similar incompatibilities arise in other dimensions of resolution such as time and space. Time-steps vary from nanoseconds to minutes. When two simulators with disparate time-steps are linked, the one with the smaller time-step may interpret the lack of response from the other as inaction when in fact, the other will report its action only at the end of its larger time-step. This could easily lead to inconsistencies. Assumptions about the resolution of terrain also vary across models. These differences can lead to inconsistencies such as tanks flying several feet above the ground.

A technique to resolve these incompatibilities is to provide *translators* between levels of resolution. In the two-level case of Figure 4, a translator is a diagonal linkage. We argue that such translators are useful only in special cases; in general, they cannot provide the effectiveness required by many linkages. We elaborate using examples for three dimensions of resolution: unit size, time-step and terrain.

Pseudo-disaggregation has been proposed to solve some of the problems associated with multiple resolutions of unit size. The basic idea is that the perceiver of an aggregate entity applies a local translation function to obtain a disaggregated view of the aggregate entity. This technique works well as long as perception is the only interaction — it fails if the perceiver also engages the perceived in combat[3] since the perceived units do not respond to events (attack, defend, retreat, etc.). To achieve a completely realistic engagement, the perceived units must respond as if they were being modeled as individual entities themselves.

Smoothing and clustering are techniques that have been used to resolve time-step differentials. Smoothing is the act of spreading the effects of an aggregated interaction (computed instantaneously) over a period of simulated time while clustering is the opposite — instantaneously simulating the effects of interactions that occurred over a period of time. While an argument can be made for the efficacy of the former, we contend that the latter will generally violate effectiveness requirements (fidelity and validity). Such violations have been reported in practice [Seidel95]. By making the time-steps compatible (in accordance with FO-1), most of these problems may be eliminated.

Finally, in the case of terrain resolution, a simple mathematical mapping function may suffice to translate coordinates between systems. However, in some situations such functions do not exist (e.g., when one model operates in two-dimensional space while the other operates in three-dimensional space). Further, the difference in resolution (e.g., meters versus kilometers) can lead to inconsistencies similar to those observed with time-step differentials.

Thus, while point-to-point translations may suffice for some linkages, entities must usually be modeled at the appropriate level or levels of resolution to achieve the required effectiveness.

Since interactions may occur at any level at any time, in order to satisfy FO-1, entities must either (i) maintain representations at all levels at all times, or (ii) dynamically transition to the appropriate level as required. The costs

3. Note, this engagement may be achieved indirectly by communicating the perceived state to a combat unit.

associated with the first approach are obviously prohibitive. Thus far, the second approach, known commonly as aggregation-disaggregation, has been adopted. As noted in Section 3.4, dynamic transitions across levels of resolution have associated overheads as well. Thus, a corollary of FO-1 is that transition overheads are a given.

In order to reduce these transition overheads, a hybrid approach may be adopted, combining elements of the two schemes mentioned above. In this approach, each entity maintains a consistent *core* of attributes, spanning the levels of resolution. The core is a subset of the entire set of attributes, consisting of only those attributes that are deemed essential. As needed, values of additional attributes are generated at any level from those in the core. A more detailed discussion of the core including criteria for attribute membership in the core is given in Section 7.1.3.

6.2 Fundamental Observation 2: Consistent Combining

Given the costs of dynamic transitions across levels of resolution, we must focus on minimizing these costs. Transition costs can be reduced in two ways: (i) by reducing the cost associated with a single transition, and (ii) by reducing the number of transitions. Here, we focus on the second — Section 7.1.3 presents an approach to reduce the cost of single transitions. Significant reductions in overhead can be achieved by limiting the propagation of transitions (for example, by controlling chain disaggregation). Ideally, a transition should be restricted to a single entity and not allowed to propagate at all. As seen in Figure 5, this leads to the following two requirements: (i) entities must be able

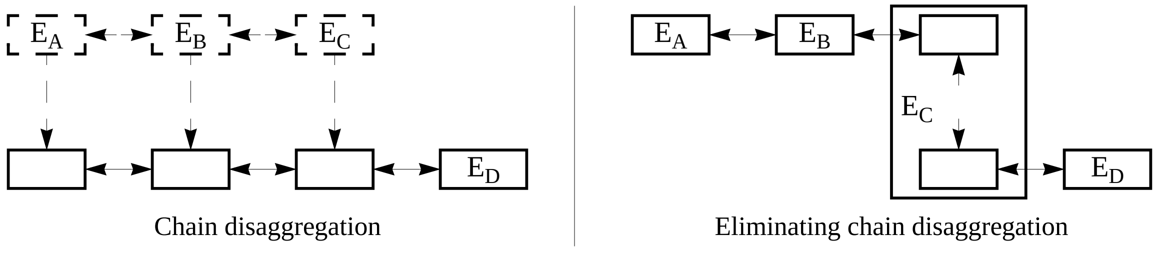

FIGURE 5: Reducing transition overheads by limiting propagation of transitions

to handle *concurrent* interactions (i.e., interactions occurring within simulated periods that overlap) at multiple levels, and (ii) the effects of these concurrent interactions must be combined without compromising effectiveness (fidelity, validity, consistency, etc.). In Figure 5, entity E_C must interact concurrently with entities E_B and E_D in order to limit the propagation of the transition. These requirements are captured in FO-2 below:

FO-2: The effects of concurrent interactions at multiple levels of resolution must be combined consistently.

The difficulty arises in satisfying the second requirement. On the one hand, interactions could be serialized, i.e., processed sequentially and atomically. This approach fails in the context of real-time interactions. Two interactions that overlap in real-time *must* appear to be executed concurrently. Serializing the transactions detracts from this appearance, especially if some transactions are longer than others.

Alternatively, interactions could be processed in parallel and their results combined. Although apparently reasonable, this approach has several pitfalls as well. The subtleties of these are best explained using an example. Consider the following scenario (Figure 6): LRE_1 and LRE_2 are two platoons of tanks, engaged in battle. At the same time, LRE_2 is also engaged with two individual tanks — HRE_1 and HRE_2. The battle between LRE_1 and LRE_2 is being simulated at the aggregate level while the battle with HRE_1 and HRE_2 must be simulated at the disaggregate level. Assume the time-step for the aggregate-level interactions is one minute. Time-steps for individual tank simulators are typically on the order of milliseconds. During a particular one-minute time-step, LRE_1 inflicts 50% attrition on LRE_2. Also, during this interval, HRE_1 and HRE_2 destroy two tanks in LRE_2[4]. Since LRE_2 has four tanks, the 50% attrition equals the destruction of two tanks. The question is, how should these two results be

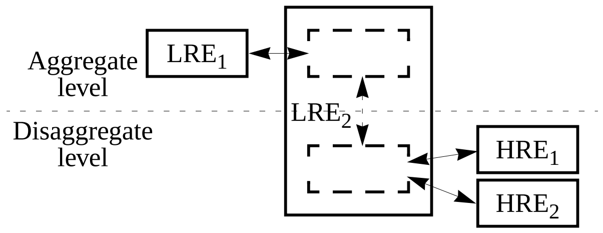

FIGURE 6: Concurrent interactions at multiple levels

[4] Typically, platoon-level engagements are specified in terms of percentage attrition, whereas tank-level engagements are specified in number of tanks lost.

combined? Depending on the amount of overlap in the two interactions, the final result could be a reduction in LRE$_2$'s strength by 50% (complete overlap), 75% (partial overlap) or 100% (no overlap). For the most part, this choice must be made arbitrarily and the result assumed to be realistic. Unfortunately, particular choices may lead to an unfair fight (see Section 5). As another example, consider a one-minute time-step in the aggregate-level interaction during which, LRE$_2$ expends 75% of its available ammunition. HRE$_1$ and HRE$_2$ also engage LRE$_2$ during this time-step, causing LRE$_2$ to expend 40% of its available ammunition. At the end of the time-step, LRE$_2$ will have expended 115% of its ammunition — a physically impossible outcome.

The problem may be characterized generally as follows: an interaction, L, operating over some time-step generally makes assumptions about the states of the interacting entities over the duration of the time-step (typically, these are based on the basic premise that L is the only interaction occurring during the time-step). When interactions (S_i) with smaller time-steps are allowed to occur concurrently with those (L_i) with larger time-steps, the assumptions made by the L_i will be invalidated due to the S_i during a time-step, leading to ineffective linkages.

6.3 Fundamental Observation 3: Dependent Concurrent Interactions

The problem of combining the effects of concurrent interactions consistently superficially appears to be tightly coupled to time-step differentials. In fact, consistency problems arise in linkages primarily due to a more fundamental underlying problem: *interaction dependence*, an interaction's existence or effects depending on another interaction.

Consider the more detailed view of Figure 6 shown in Figure 7. Assume equal time-steps. In some time-step t_i, the interaction between LRE$_1$ and LRE$_2$ results in LRE$_2$ reducing the ammunition of a constituent tank (T) by 25%. In effect, T has fired at LRE$_1$ during t_i. Also, in t_i, the disaggregate-level interaction between LRE$_2$ and HRE$_1$ involves T firing at and damaging HRE$_1$. Thus, both interactions involve the firing of a weapon by T *in the same time-step*. Clearly, this is physically impossible (indicated in Figure 7 by tank T having two turrets). By allowing such an outcome, the simulation allows an unfair engagement. Thus, assuming compatible time-steps does not eliminate inconsistencies. The problem lies in the fact that two interactions that occur in overlapping simulation times involve a common entity, and thus affect each other's outcome.

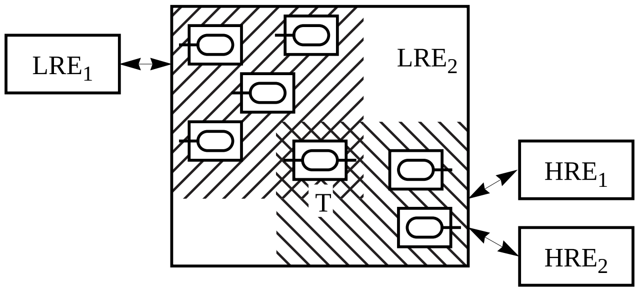

FIGURE 7: Dependency considerations

FO-3: Concurrent interactions may be dependent.

Two interactions that overlap in (i) simulation time, and (ii) the entities involved in the interaction, may not in general be independent simply because they can affect the outcome of each other. In the worst case, as in the example of Figure 7 above, one might preclude the other. If two interactions that are dependent are executed independently, effectiveness will be compromised when the results of these interactions are combined.

Returning to the example of Figure 7, the two interactions of interest are the aggregate-level interaction between LRE$_1$ and LRE$_2$ (call it I_1), and the disaggregate-level interaction between tank T in LRE$_2$ and HRE$_1$ (call it I_2). I_1 and I_2 are executed independently. Unfortunately, both of them involve the firing of a weapon by tank T. Since T can fire only in one of the two interactions, I_1 and I_2 are in fact, dependent. Therefore, the results generated by executing them independently are ineffective.

6.4 Fundamental Observation 4: Time-step Differential

In Section 6.3, we have shown that the fundamental issue underlying consistent combination of concurrent interactions is dependence among transactions and not differences in time-steps. However, time-step differentials do play a role in this problem — they tend to aggravate the inconsistencies created due to dependency issues. Thus,

FO-4: Time-step differentials can amplify ineffectiveness due to dependence violations.

Recall ineffectiveness can occur when dependent interactions are executed independently. A necessary condition for two concurrent interactions to be dependent is overlap in simulation time. Thus, the greater this overlap, the higher the potential for ineffectiveness. Figure 8a shows the overlap in simulation time due to two equal time-steps. In this case, the overlap is limited to two time-steps (and hence two interactions). When the time-step of the second simulation is reduced by a factor of five (Figure 8b), the overlap increases to six time-steps (interactions). Thus, the larger the time-step differential, the larger the number of concurrent interactions, and potentially, the greater the ineffectiveness of the linkage.

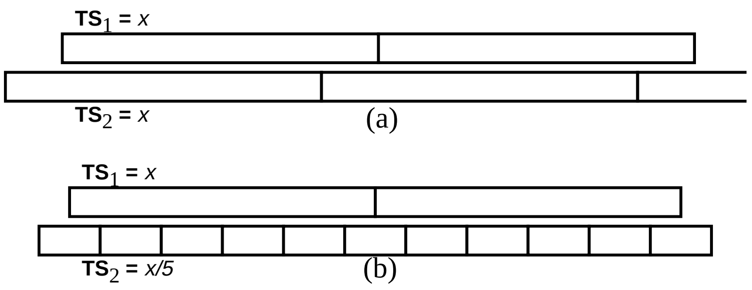

FIGURE 8: Time-step differential increases overlap

When dealing with linkages between legacy simulations (such as `AWSIM/ModSAF`, `Eagle/BDS-D` and `BBS/SIMNET`), time-step differentials are a reality. Disaggregate simulators (such as `SIMNET` tanks) operate at the millisecond time-step level. On the other hand, aggregate-level interactions typically use equations with coefficients derived from historical data aggregated over periods ranging from several minutes to days [Karr83, Epst85]. Hence, time-steps of several minutes to a few hours are typical for aggregate-level interactions. Resolving the time-step differential appears to be a very difficult problem, especially for legacy systems. FO-4 suggests that future simulation efforts must be directed towards solving this problem if effective cross-resolution linkages are to be achieved.

6.5 Fundamental Observations Summary

The fundamental observations present the basic issues that must be addressed by any general, scalable approach to cross-resolution modeling and thus provide the beginnings of a theoretical foundation for the same. The key to cross-resolution modeling is a holistic approach that internalizes issues of consistency and is designed to solve them. In the next section, we extend our theory by presenting one such approach based on the fundamental observations.

7 A Unified Approach to Cross Resolution Modeling

Traditional approaches towards aggregation-disaggregation maintain for a given entity, at any given time, the attributes at only one level of resolution — the level at which the entity is being simulated. This is unsatisfactory because when an entity is simulated at a certain level of resolution, attributes at other levels are unused or lost. Typically, the entity is ghosted at the levels at which it is not simulated. However, ghosting implies a passive reflection of attributes, not a participation at multiple levels. From FO-1, for an entity to actively participate in interactions at multiple levels, it must not only be influenced by events from other entities, but also influence other entities with events at multiple levels. An entity that does so could be said to exist at multiple levels of resolution.

While existing solutions may work for specific cases of two simulations being linked, they cannot ensure effective linkage in general. Our approach to cross-resolution modeling is a general one in that the focus is on the maintenance of consistency based on the fundamental observations of Section 6. The focus of our approach is maintenance of consistency among multiple levels of resolution. We propose Multiple Resolution Entities (MREs) for maintaining internal consistency across multiple levels of resolution. MREs reflect a concept, *not* an implementation. MREs may be designed during the construction of a multi-level simulation or may be created by linking together existing simulations with suitable changes made to incorporate the concepts discussed here.

7.1 Multiple Resolution Entity (MRE)

In concept, an MRE interacts at multiple levels of resolution concurrently by internally maintaining consistency among corresponding attributes at different levels of resolution. Each MRE either maintains state information at all desired levels of resolution or furnishes information at a requested level in a timely manner. Simulation of the MRE entails reflecting the effects of incoming interactions at all desired levels. Each MRE is responsible for enforcing logical consistency across resolution levels; the effect of any incoming interaction should be reflected consistently in the attributes of all levels of the MRE. Figure 9 depicts a typical MRE that can be perceived at two levels of resolution. For the sake of discussion, $Level_1$ is the low-resolution level and $Level_0$ is the high-resolution level. The MRE maintains the attributes at both levels at all times (in Section 7.1.3 we suggest alternatives to storing all the attributes at all levels at all times). The two states of the MRE — $Level_0$ and $Level_1$ states — are always kept consistent with each other. Let E_1 be an MRE, E_2 be a $Level_1$ entity and T be a $Level_0$ entity. Interactions between E_1 and E_2 occur at $Level_1$. T requests $Level_0$ information when it comes into contact with E_1. E_1 proceeds to send information regarding T_{1-4} to T. This information would be culled from data the MRE maintains on each of T_{1-4}. E_2 receives information sent from E_1's "global" fields — the fields that are either common to T_{1-4} or can be deduced from the individual T_i attributes.

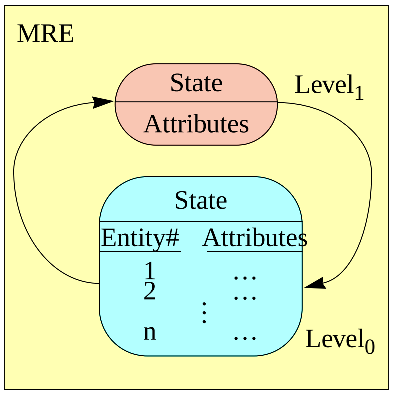

FIGURE 9: Design of an MRE

Figure 10 shows how the MRE concept could be applied to a practical cross-resolution linkage, such as `CLCGF` (Corps Level `CGF`). This application consists of `Eagle`, a $Level_1$ model, and `ModSAF`, which provides computer-generated forces at $Level_0$. The two models would maintain attributes and model behaviors at their respective levels, while the "Consistency Enforcer" would be one or more pieces of software responsible for maintaining consistency across the two models. For example, the Consistency Enforcer could include strategies for combining the effects of incoming interactions in a single time-step, distributing these effects across the two levels, mapping the effects of interactions at one level to the attributes at the other, etc. A detailed design for the Consistency Enforcer would depend on the particular models chosen at the various levels and is beyond the scope of this paper. Although `Eagle` and `ModSAF` are separate pieces of software, as an MRE, they are designed to act like a single, coherent entity (E_1) that can interact at multiple levels concurrently. Another aggregate model (E_2) such as another instantiation of `Eagle`, could engage the `Eagle` model in E_1. Simultaneously, a crewed tank simulator T could engage the tanks T_{1-4} in E_1. It is the function of the Consistency Enforcer to ensure these concurrent interactions are handled consistently.

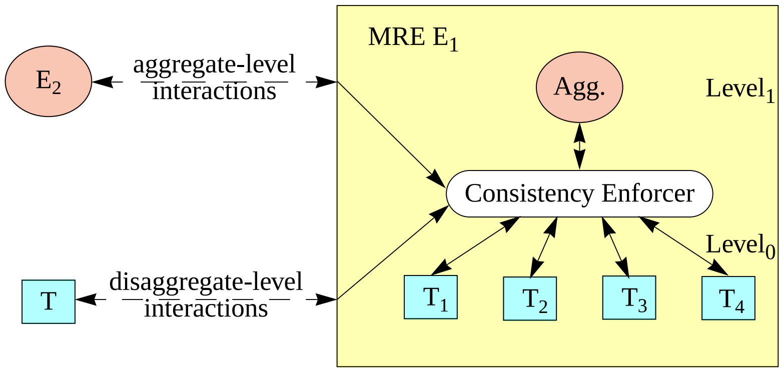

FIGURE 10: Multiple levels of resolution

7.1.1 Concurrent Interactions

FO-2 notes that for entities to interact at different levels of resolution, concurrent interactions at multiple levels must be addressed. Also, the effects of these interactions must be consistently reflected at all levels. We believe that the solution to concurrent interactions at multiple resolutions lies in designing entities to be able to process these interactions. Traditional approaches to this problem have been more in the nature of point-to-point solutions (see Section 4), wherein the issues are addressed for the specific case of two interacting simulations. This neither leads to a general solution, nor does it guarantee that the solution will be scalable for even those two simulations.

We propose the MRE as an entity that is able to reflect the effects of concurrent interactions at multiple levels in a consistent manner in accordance with FO-2. A key aspect of the MRE is that it internalizes the consistency maintenance that is essential to solving the issues outlined earlier. The MRE consistently reflects in its attributes the effects of interactions at all allowed levels of resolution. By being able to furnish the attributes at any level in a timely manner on demand, the MRE is capable of being perceived at multiple resolutions at overlapping simulation times. By maintaining internal consistency, the MRE can ensure that the multiple views are all consistent with each other.

7.1.2 Consistency Maintenance

Consistency maintenance among the levels of resolution is the crux of CRM. There are two aspects to consistency — temporal and mapping. Temporal consistency involves reconciling the effects of the interactions at different levels in a manner such that the MRE presents consistent views at different levels. Mapping consistency pertains to designing functions that are needed to map attributes from one level to the other. We call these functions mapping functions. A two-level MRE may require a pair of functions f and f^{-1} such that f maps a set of attributes at the disaggregate level to a set at the aggregate level while f^{-1} maps attributes from the aggregate level to the disaggregate level. If more than two levels of resolution exist, then mapping functions must be found for each pair of levels that can transition from one to the other. An important property of mapping functions is that they must be reversible. For example, a mapping function f that translates disaggregate attributes to an aggregate equivalent must have a dual f^{-1} that translates the aggregate attributes to the disaggregate consistently.

7.1.3 Core

The issue of how the MRE maintains consistency internally is still an open one. The MRE approach as outlined here is intended only to serve as the foundation for consistency maintenance techniques. Much work remains before any conclusions can be drawn about the efficacy of such techniques. We present an approach that addresses transition latency and mapping consistency issues, as depicted in Figure 11.

Each MRE maintains a set of attributes at all times from which it can generate all attributes at all desired levels of interaction in a timely manner on demand. This set of attributes — the core set or core — may be updated on every interaction to reflect a state of the MRE that is consistent at multiple levels of resolution. Conceptually, the core could be comprised of all the attributes at all the levels of resolution. Thus, if a disaggregate-level interaction occurs, its effects can be reflected at even the aggregate level by employing the appropriate mapping functions. However, we envisage the core set as being a subset of all the attributes. Attributes in the core would be chosen such that they may be fleshed out to all the attributes for the level simulated. The core set would be maintained at all times in the simulation, but the other attributes may be discarded when they are no longer needed. Some attributes may be "aged" to reflect decreasing utility with time. As such, the core could help reduce transition latency.

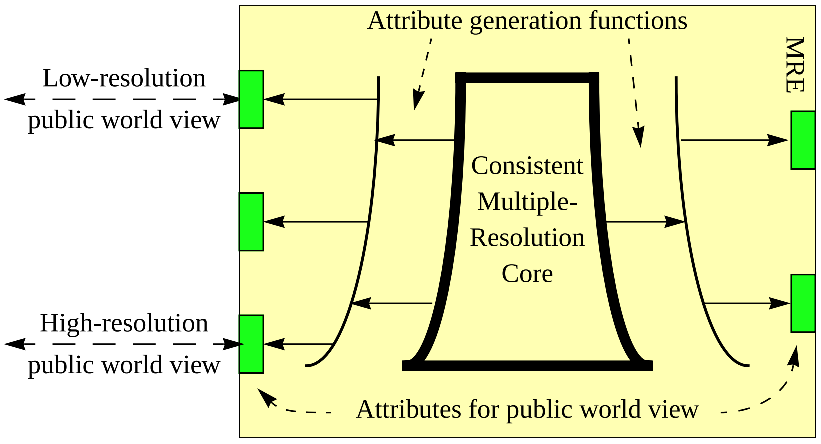
Attribute generation functions
MRE
Low-resolution
public world view
High-resolution
public world view
Consistent Multiple-Resolution Core
Attributes for public world view

FIGURE 11: Core attributes

Since the core is a subset of all the attributes at all levels, we need to develop criteria that would identify attributes that should be in the core. We have identified four criteria that should be considered when attempting to identify core variables.

- Reversibility
- Decreasing validity with time
- Cost ratio
- Frequency of access

Reversibility: For many attributes, it is important that there exist mapping functions that translate the values at one level to the values at another level. Also, this translation must be done in such a manner as to retain all information so that a reverse translation can be done. However, in many cases these mapping functions may be hard to find or encode. In such cases, when the attributes require reversibility but mapping functions cannot be found, the attributes must be included at all levels of resolution in the core. This ensures that the attributes are current and relevant for their respective levels. Consider an application for which the position attribute requires reversibility but mapping functions cannot be found. The position of the aggregate may be found out by averaging the position of the disaggregate entities. Likewise, a doctrine or template may be applied to the aggregate position to determine disaggregate position. However, these functions are relevant only when the entities are not perturbed by other interactions. If the positions of the disaggregate entities change by small amounts due to disaggregate-level interactions, then it would not be possible to generate those new positions from the aggregate position. In such a case, perfectly reversible mapping functions cannot be found and hence position attributes must be stored at both levels. Note that this is a very specific example. A counter-example wherein it is not necessary to store position attributes at all levels may be for a molecular simulation where if the position and orientation of a molecule is known, it is not necessary to store the positions of the individual atoms.

Decreasing validity with time: Another criterion is whether the attribute's validity decreases or not with time. Obviously, the attribute could be kept in the core when it is useful and when its validity goes below a threshold it could be removed from the core.

Cost ratio: Cost ratio is the ratio of the cost of maintaining the attribute to the cost of generating it. If the cost of maintaining the attribute is measured by the amount of memory it consumes and the cost of generating it is measured by the time it takes to generate it, then this criterion reduces to a space-time trade-off. On the other hand, the cost of maintaining the attribute could be measured by the amount of time required to change its value, in which case the comparison lies between the time to effect a change and the time to generate the attribute. Obviously, whether the attribute should be maintained in the core or not depends on this ratio being larger than, smaller than or equal to one.

Frequency of access: Our fourth criterion is the frequency with which the attribute is accessed. If the frequency is high, then it may be judicious to incorporate the attribute into the core.

Note that all the abovementioned criteria are quite independent of each other. It may happen that for some applications, the criteria may conflict with each other. In such a case, appropriate weights should be assigned to the criteria to aid selection of the core attributes.

The concept of the core is quite orthogonal to the concept of the MRE. The MRE advocates internal consistency and addressing concurrent interactions at multiple levels. The core is one method of maintaining internal consistency, but by no means the only method.

7.2 Practical Considerations

Over the years, substantial investment has been made in producing simulation programs that are unfortunately incompatible with each other. ALSP (Aggregate-Level Simulation Protocol) [Weat93] presents a framework for linking simulations at different resolution levels. However, a large number of simulations were intended to be stand-alone and continue to be so. Increasingly, the view is that different simulations should be able to work together [DIS93]. There have been two initiatives towards this goal. One has been to make existing simulations — legacy simulations — work together. The second has been to devise standards for all future simulations, wherein interoperability is a requirement and not an afterthought [DoD94]. Although the framework presented is better suited to the second initiative, it is not incompatible with linking legacy simulations. As described in the discussion associated with Figure 10, we believe a two-level MRE could be constructed (for example) using an aggregate simulation such as `BBS`, a disaggregate simulation such as `ModSAF` and new components designed to arbitrate the maintenance of consistency between the two.

The MRE concept is still evolving. It is a way of forcing designers of multi-resolution models to focus on internal consistency and design it into the cross-resolution linkage. Due to the variety of simulations to which the MRE concept could be applied, a general theory will be limited. We envision the MRE concept being the basis of a general design framework to facilitate construction of consistent cross-resolution linkages.

8 Conclusions

The Cross-Resolution Modeling (CRM) problem poses significant challenges, particularly with respect to maintaining consistency among multiple, concurrent levels of resolution of the same abstraction. We have attacked the CRM problem by offering a set of four fundamental observations that narrow the problem considerably. Also, these observations point to changes that should be implemented in all future attempts at CRM; past attempts have failed to appreciate most if not all of these newly-discovered observations, generally with undesirable results in the effectiveness of the resulting simulations.

Our fundamental observations narrow the CRM problem but do not solve it. The primary remaining issue concerns consistency maintenance in multiple, concurrent levels of resolution of the same abstraction. How to solve this key problem is still itself an open issue. At this point we believe the answer will invariably be application dependent. We mentioned earlier the flexibility in accuracy that is generally available while still meeting the requirement for a fair fight. The degree of flexibility is what appears to be quite application dependent.

We have proposed MREs as a method such that if consistency can be realized satisfactorily, MREs are the approach that will enable that realization. Past attempts at the CRM problem have been truly *ad hoc* in concept and in implementation. MREs can help with conceptual organization and will greatly aid ease of implementation. Also, MREs, by their nature, enable efficiency enhancements. We have discussed the notion of a core set of attributes and the conditions under which they should be maintained. This idea of a core set is one of the most promising efficiency enhancements we have seen. Time will tell how well the notion of a core set works out in practice.

Future work addresses the consistency issue. We have explored it in some depth, but many avenues remain. An avenue that holds significant potential for success is application of VRM ideas to the CRM problem. We believe VRM should be pursued aggressively as the key part of any solution to the CRM problem. We have formulated *Attribute Dependency Graphs* (ADGs) as a means of capturing the dependencies between attributes at different levels of resolution. In addition, we have performed a cost analysis of some existing CRM techniques and compared them with ours. We expect to formulate a framework comprising of MREs, ADGs, the cost analysis and other as-yet-unclear tools in order to provide consistency maintenance guidelines to designers of multi-level simulations.

9 References

[AMG95] Architecture Management Group, *Preliminary Definition*, High Level Architecture Briefings, Defense Modeling and Simulation Office (DMSO), Alexandria, Virginia, March 31, 1995.

[Burd95] Burdick, C. D., Loral, *Interoperability of Simulations with Different Levels of Resolution*, Defense Modeling and Simulation Office (DMSO) Workshop, Alexandria, Virginia, November 29-30, 1995.

[Calder95] 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*, Proceedings of the 5th Conference on Computer Generated Forces & Behavioral Representation, Orlando, Florida, May 1995.

[Clark94] Clark, K. J., Brewer, D., *Bridging the Gap Between Aggregate Level and Object Level Exercises*, Proceedings of the 4th Conference on Computer Generated Forces & Behavioral Representation, Orlando, Florida, May 1994.

[Cox95] Cox, A., Maybury, J., Weeden, N., *Aggregation Disaggregation Research — A UK Approach*, Proceedings of the 13th DIS Workshop on Standards for the Interoperability of Distributed Simulations, Orlando, Florida, September, 1995.

[Davis93] Davis, P. K., Hillestad, R. J., *Families of Models that Cross Levels of Resolution: Issues for Design, Calibration and Management*, Proceedings of the 1993 Winter Simulation Conference, 1993.

[DIS93] DIS Steering Committee, *The DIS Vision, A Map to the Future of Distributed Simulation*, Comment Draft, October 1993.

[DoD94] Under Secretary of Defense (Acquisition and Technology), *Modeling and Simulation (M&S) Master Plan*, Dept. of Defense, September 30, 1994.

[Epst85] Epstein, J. M., *The Calculus of Conventional War: Dynamic Analysis Without Lanchester Theory*, The Brookings Institute, 1985.

[France93] Franceschini, R. W., *Intelligent Placement of Disaggregated Entities*, Institute for Simulation and Training, 12424 Research Parkway, Suite 300, Orlando FL 32826.

[Hardy94] Hardy, D., Healy, M., *Constructive & Virtual Interoperation: A Technical Challenge*, Proceedings of the 4th Conference on Computer Generated Forces & Behavioral Representation, Orlando, Florida, 1994.

[Karr83] Karr, A. F., *Lanchester Attrition Processes and Theater-Level Combat Models*, Mathematics of Conflict, Elsevier Science Publishers B.V. (North-Holland), 1983, ISBN: 0 444 86678 7.

[Petty94] Petty, M. D., *The Turing Test as an Evaluation Criterion for Computer Generated Forces*, Proceedings of the 4th Conference on Computer Generated Forces & Behavioral Representation, Orlando, Florida, May 1994.

[Robkin92] Robkin, M., *A proposal to Modify the Distributed Interactive Simulation Aggregate PDU*, Hughes Training, Inc., February 28, 1992.

[Seidel95] Seidel, D. W., King, B. C., Burke, C. D., *AIM Approach to Simulation Interoperability*, The MITRE Corporation, Preliminary Draft, July 7, 1995.

[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*, Proceedings of the 5th Conference on Computer Generated Forces & Behavioral Representation, Orlando, Florida, May 1995.

[Turing50] Turing, A. M., *Computing Machinery and Intelligence*, Mind, volume 59, 1950.

[Weat93] Weatherly, R. M., Wilson, A. L., Griffin, S. P., *ALSP - Theory, Experience and Future Directions*, Proceedings of the 1993 Winter Simulation Conference, 1993.