Chapter 3
Intelligent Entity Behavior Within Synthetic Environments

***

R.V. Kruk, P.B. Howells, and D.N. Siksik
CAE Inc.
St. Laurent, Quebec, Canada

Abstract

This paper describes some elements in the development of realistic performance and behavior in the synthetic entities (players) which support Modeling and Simulation (M&S) applications, particularly military training. Modern human-in-the-loop (virtual) training systems incorporate sophisticated synthetic environments, which provide:

1. *The operational environment, including, for example, terrain databases;*
2. *Physical entity parameters which define performance in engineered systems, such as aircraft aerodynamics;*
3. *Platform/system characteristics such as acoustic, IR and radar signatures;*
4. *Behavioral entity parameters which define interactive performance, including knowledge/reasoning about terrain, tactics; and,*
5. *Doctrine, which combines knowledge and tactics into behavior rule sets.*

The resolution and fidelity of these model/database elements can vary substantially, but as synthetic environments are designed to be compose able, attributes may easily be added (e.g., adding a new radar to an aircraft) or enhanced (e.g. Amending or replacing missile seeker head/ Electronic Counter Measures (ECM) models to improve the realism of their interaction).

To a human in the loop with synthetic entities, their observed veridicality is assessed via engagement responses (e.g. effect of countermeasures upon a closing missile), as seen on systems displays, and visual (image) behavior. The realism of visual models in a simulation (level of detail as well as motion fidelity) remains a challenge in realistic articulation of elements such as vehicle antennae and turrets, or, with human figures; posture, joint articulation, response to uneven ground. Currently the adequacy of visual representation is more dependant upon the quality and resolution of the physical models driving those entities than graphics processing power per Se.

Synthetic entities in M&S applications traditionally have represented engineered systems (e.g. aircraft) with human-in-the-loop performance characteristics (e.g. visual acuity) included in the system behavioral specification. As well, performance—affecting human parameters such as experience level, fatigue and stress are coming into wider use (via AI approaches) to incorporate more uncertainty as to response type as well as performance (e.g. Where an opposing entity might go and what it might do, as well as how well it might perform).

1. Introduction

Synthetic environments play a major role in modern military training systems. Over the years they have progressed from providing basic threat layout to incorporation of the cooperative elements necessary for collaborative training in land, sea and air applications. To a large extent, this has been made possible through advances in computer-generated simulation of military forces using modeling technology, networking capabilities and improved computer hardware [6].

Modern synthetic environments provide solutions for a wide range of training needs. Using knowledge-based systems and detailed mathematical modeling of physical systems, it is possible to build simulations where the manned simulator works in concert with the computer-generated entities. A simulator may be linked to various synthetic environments via networking protocols such as the High Level Architecture (HLA) [3], allowing the simulator to see and be seen by the computer-generated entities. Direct communications between simulator and entity may be achieved via the simulation of digital radio and messaging, thus enabling a trainee to request data updates from and submit direct requests to computer-generated allies.

In the aviation arena, there is a great deal of emphasis on air-to-air combat mission training. This is because aircraft very rarely go into combat alone. The computer-generated entities play an important role in providing a manned simulator with wingmen and/or coordinated opponents.

In the naval arena, synthetic environments provide for training in anti-submarine warfare (ASW), anti-surface warfare (ASuW) and search and rescue missions. Naval tactics such as screening and tracking coupled with classification strategies for naval contacts (naval identification criteria) as well as Link 11 communications capability combine to support multiple mission exercises, a role fulfilled by the computer generated entities in modern day training devices.

Within the ground forces environment, simulation training revolves around the battle, command and control, geography, environment and logistics for relatively large numbers of individual entities, which may be aggregated in a number of ways. Synthetic environments provide the terrain, tactics, allied forces, opponents and weather elements necessary for realistic friendly and hostile interactions.

2. A Note on Standards

An extensive range of standards has grown up around the definition of platform and system characteristics in Networked Military M&S. These include specification of how platform and system entities are to be accessed, which began in the development of Protocols for Distributed Interactive Simulation (DIS) and have been carried over into the High Level Architecture protocols [4, 8]. Within HLA, further entity specification is defined in the Object Models for Simulations (SOMs) and Federations (FOMs). Performance levels of models for aircraft, radars, ordinance, etc., can be clearly defined, are predictable, and the implementation in simulation at run time is deterministic. This is not the case for representation of human entities (such as individual soldiers).

There are currently no widely accepted standards for representation of high fidelity human entities in simulation. Human models should be generated and run through approaches fundamentally different from those used for simulation of engineered constructs such as vehicles - if for no other reason, because of the necessity to represent far more degrees of freedom in motion, as well as model individual and aggregate behavior which may include emotional effects and "Situation Awareness". Traditionally human behavior in large command and control training systems has been generated by role—playing human operators—in some cases 50 role players might support the training of a 10 person command team. Automating these characteristics in Synthetic Environments used for training remains a significant challenge to system designers.

3. Some Examples of Human Modeling/Simulation Problems

1. Slippage/penetration: A synthetic human's feet and hands appear to slip on surfaces as if on ice or go through surfaces because knowledge about the precise location of the surface is not inherent in the Human joint model—rates of motion and degrees of freedom do not match, and inherent flexibility of human torso/limbs is difficult to simulate.

2. Inability to climb objects and/or deal with minor terrain variation: Same problem set as 1, plus difficulty in defining borders of 3D objects in conventional simulation architectures.

3. Jerkiness: A synthetic human jerks in transitions between postures because of lags between limb activation, difficulties in representing joint degrees of freedom and flexibility (especially the spine) and balance is anomalous (synthetic entities often look like they are about to fall over).

4. Inability to grasp objects: Same problem sets as 1 to 3—hands are often modeled like clubs.

We are able to model and simulate humans within machines much more effectively, and the bulk of the content of the present paper concentrates on the state of the art for humans embedded within systems.

4. General Architecture

The general architecture of a simulation application is an assembly of frameworks, libraries and data organized in layers as shown in
Figure *1*. The framework that is the foundation of the architecture is the Virtual Environment Framework. It defines how all the pieces fit together and interact with each other [2,7]. The next layer is composed of the application frameworks. Each one of these frameworks isolates a particular domain to allow various implementations to be used. The relationship between these frameworks should be limited to interfaces, which define the functionality, but not the implementation. The third layer is a set of libraries provided by various manufacturers. They provide an implementation of their associated framework. The last layer is composed of initial values for the objects defined in their associated library. All these data are linked together within the scenario to be executed in real-time.

Figure *1* also presents a breakdown of application frameworks that are related to the synthetic tactical environment domain. The main framework of this domain being the computer generated forces. The other frameworks either extend the capabilities (an expert system framework to give realistic behavior to players) or provide complementary capabilities to increase the realism of the simulation (terrain and weather frameworks).

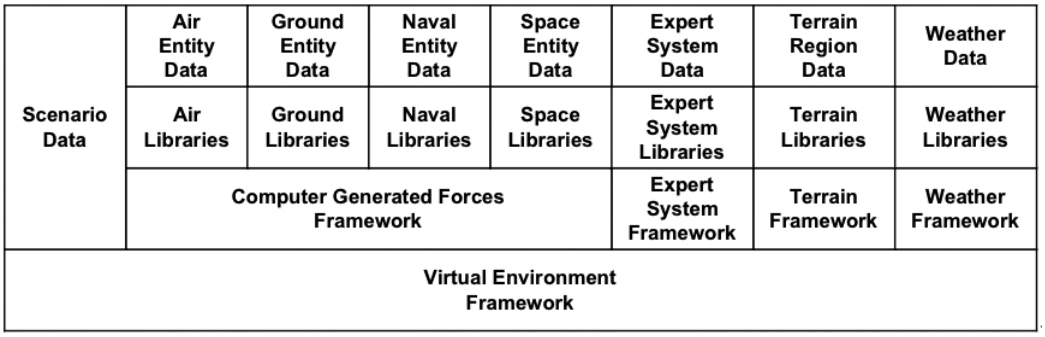

Figure 1. General Architecture of a Simulation Application

5. Virtual Environment Frameworks

The virtual environment framework implements the infrastructure of the general architecture. It is internally divided into three layers that provide various functions, as shown in Figure *2*.

The first layer, i.e. the foundation layer, includes the services necessary to support a truly expandable and configurable system. At the implementation level, it defines the plug-in concept to support libraries and other frameworks, and the data type concept to support attributes and object factories [1]. At the functional level, it defines the provider concept for adding global functionalities and the adapter concept for adding functionalities to individual objects. It also defines the interface with the repository and provides the default implementation.

The second layer, i.e. the execution layer, handles the publish, subscribe and ownership mechanisms. It defines interfaces with the scheduler and with the network. It also provides the default implementations for both. Within the context of this layer, each object performs its own publish and subscribe operations. The framework combines these operations before making similar operations through the network interface. An additional service allows access to the internal object information for verification and validation purposes.

The top layer, i.e. the distribution layer, defines how scenarios are distributed across multiple processes, and which may be executed on one or more computers. Each scenario includes initial parameters for their configurable elements of the framework such as the scheduler. When executing, the scenarios become exercises.

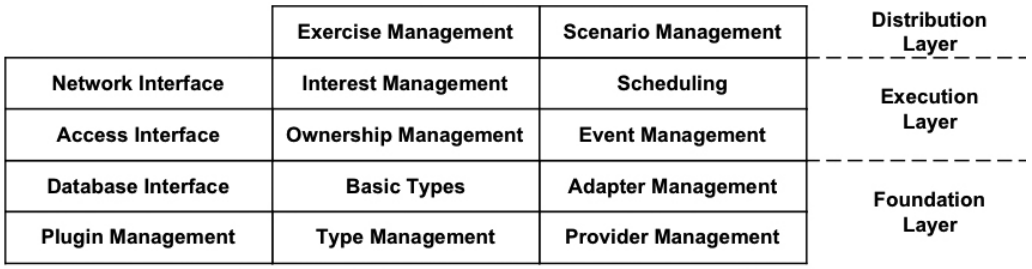

Figure 2. Virtual Environment Framework

6. Synthetic Environment (SE) Application Example 1

Air-to Air Combat with a legacy SE Tool: the Interactive Tactical Environment Management System (ITEMS) [5]

Maintaining a high degree of readiness in air-to-air combat operations is difficult. One alternative is to supplement airborne air combat training with simulator time. Effective training, however, requires that wingmen and opponents have representative characteristics and performance in terms of maneuvering, aircraft flight dynamics and decision processes.

Training can be supported in the areas of weapons deployment, basic fighter maneuvering, two-vs.-one and one-two-two engagements. For weapons training, the air target can be programmed to perform flight paths composed of simple weaves and turns, where the student is required to track the target and obtain a firing solution using guns or missiles. Control over the "g" capability of the target enables the instructor to intervene to make the task more or less difficult for the student. The simple maneuvers may be used for training in basic fighter maneuvering. The air target, for example, may be set to perform a pursuit maneuver which the manned simulator has to counter.

To support training in air-to-air combat at a more sophisticated level, ITEMS incorporates Interactive Air Targets (IATs). The air target is controlled by a programmable rule base. The rules are defined using condition and response parameters selected from a pull-down menu. For example, the condition parameters for the IAT to perform a guns engagement are for the opponent to be in the front hemisphere and in guns range. The response parameter in this case would be to fire the gun. The air target responds continuously to the rules that are triggered as the air target switches from offensive to defensive positions depending on the engagement geometry.

The air target considers the full equations of motion in the longitudinal axis with some simplifications in the lateral axis. The targets are defined using an off-line Database Management System (DBMS) for the entry of the principal aerodynamic data (lift and drag) and characteristics of the propulsion system. A range of weapon systems and sensors can be assigned to the aircraft. Typical aircraft types supported include F4-Phantom, Tornado, MiG-29 Fulcrum, and F15-Eagle, etc.

7. Components

a) Overview

The method of defining a tactical scenario is a bottom-up approach, where the elementary element databases (weapons, sensors, and countermeasures) of the scenario are first defined. These elements are then combined to generate higher-level databases that allow for a complete tactical environment by combining all elements and pertinent data. Access to the library of databases is provided via dedicated graphical user interfaces.

The principal components are illustrated in Figure 3. They include the air target definition, air combat maneuvering database, rules database, the weapons and sensor database, and the environment database. The function of each of these components is described below.

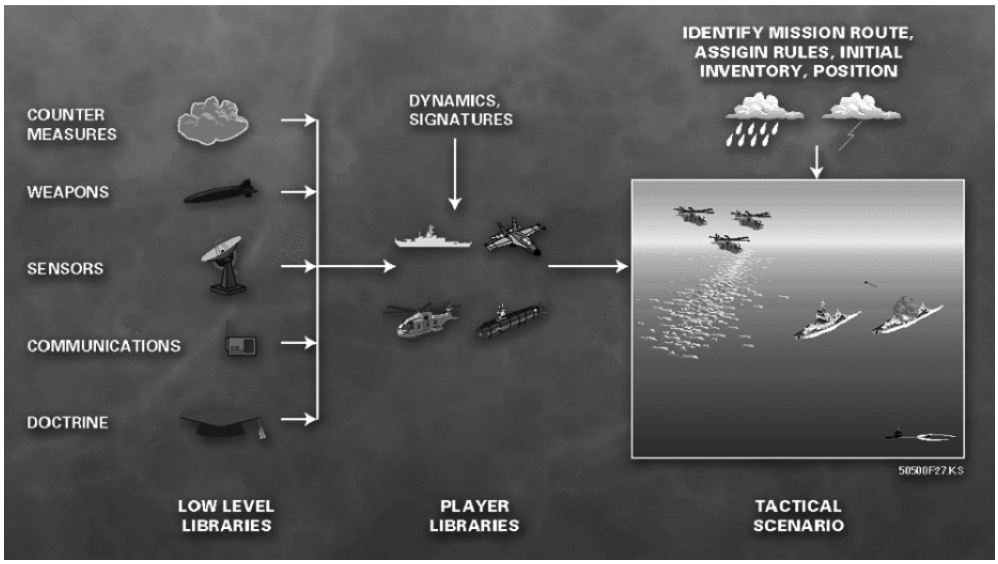

Figure 3. ITEMS Architecture

b) Air Target Definition

Typical parameters include lift and drag coefficients, engine thrust characteristics, mass and wing area.

c) Rules Database

Each player in a scenario is assigned a rule set. Usually there are two: a prime opponent selection rule set and a maneuver rule set. The opponent selection rule set contains the rules for selecting an opponent, e.g., should fast moving jets take priority over slower moving transport aircraft. The maneuver rule set is the decision kernel that invokes the maneuvers from within the air combat maneuvering database mentioned above. It is the rules defined here that instruct the air target to maneuver either offensively or defensively. The logic to transition from one maneuver

to another is also part of this rule base, i.e., when to switch from pursuit to gun track and finally disengagement.

d) Air Combat Maneuvering Database

An air combat database comprises a series of maneuver routines that can be invoked by the rule base. The rule base tells the system which maneuver to perform and the air combat database carries out that maneuver. Supported are upwards of fifteen different maneuvers to include High "G" Barrel Roll, Immelmann, Pure Pursuit, Gun Tracking, Lift Vector Turn, Scissors, Jinks, Break, High Yoyo, and Low Yoyo. These maneuvers cover the complete spectrum of offensive maneuvering, defensive maneuvering and stalemate.

e) Weapons and Sensors Database

The weapons and sensors database is used for specifying the characteristics of the weapons and sensors. Physical models consider the principal factors that affect performance. For example, in the case of the missile the user would be requested to specify the mass and aerodynamic characteristics together with the guidance mode. Countermeasures are defined in a similar manner with specification of radar cross section in the case of chaff and intensity in the case of IR flares.

At scenario load time, the IAT parameters are loaded into memory. At run-time, the aerodynamic coefficients and engine thrust rating data are interpolated based on aircraft state.

The motion path and attitude of the IAT is derived from the applied forces and moments illustrated in Figure 4, which appear as parameters in the Euler equations of motion. As shown in the figure, there are contributions from the effects of gravity, aerodynamic forces and the propulsion system. The axis systems employed include the stability axis, in which all aerodynamic forces are calculated, the body axis and the inertial axis.

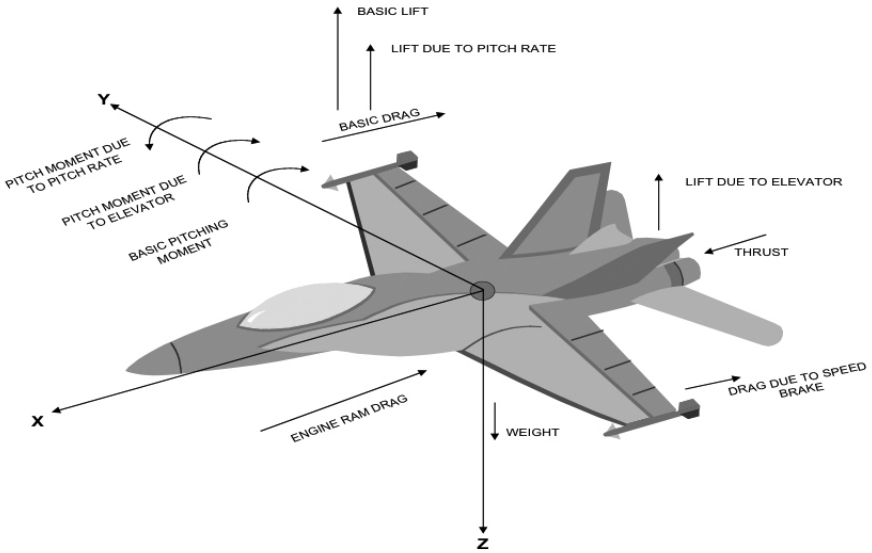

Figure 4. Aircraft Applied Forces and Moments

8. Air Target Maneuvers

Initially, the air combat rules system selects a maneuver from the maneuver list. When the selection is made, the maneuver routine proceeds to compute the position of the control surfaces and throttle position. The resulting aerodynamic loads and thrust are used to maneuver the aircraft to the desired trajectory, resulting in such maneuvers as the pursuit, yoyo, and hard turn.

The control parameters are used by the rules system to select a maneuver from the maneuver list. The list of the condition parameters is unlimited, as the user can always create new parameters as needed. The most commonly used condition parameters are those that relate to the Prime Opponent (PO), typically: Slant range, Angle Off Tail (AOT), Heading Crossing Angle (HCA), closure rate and relative height.

9. Doctrine for Air Combat

The creation of lifelike player behaviour and reaction requires the modelling of player intelligence and this, in turn, is based upon knowledge of player tactics (military doctrine). Tactics, whether used in air, ground or naval applications, require a specialized range of expertise.

Within the SE expert system, each instance of tactical knowledge is represented by IF/THEN rules and is called a “doctrine”. This knowledge provides control over the actions of individual players as well as over the summary actions of groups, such as a change of formation. Doctrines, like player data, are organized into libraries within the DBMS and are referenced by players within the scenario. Examples of doctrines include:

(1) Mission Doctrine: Knowledge pertaining to the mission of a player (goals, routes, contingencies, etc.).

(2) Prime Opponent Selection Doctrine: Criteria for the selection of an opponent for the player in question.

(3) Air Combat Doctrine: Pilot level knowledge controlling the selection of manoeuvres and weapons during air combat.

(4) Command and Control Doctrines: Doctrines applied to players organized into command structures such as flights and strike packages. These doctrines include coordination and control functions such as the cooperation of players in a package towards a common goal and the control/change of formations based on situation.

10. Proficiency

The instructor facility provides for a selection of proficiency levels that give the pilot/aircraft entity various levels of performance when maneuvering. This particular system supports Expert, Intermediate and Novice. Expert aircraft will fly at the sustainable turn and use excess energy in the vertical plane. Novice will not maintain its altitude in a high "G" turn. The Intermediate level is between the Expert and the Novice. Proficiency is built into the maneuvers and the rules sets.

11. Weapons Handling

Weapons handling for the IAT is dominated by the rules. The rules select the weapon type based on the current aircraft state. If the rule is satisfied, the weapon will be selected and the weapon fire request is granted. The IAT aiming module computes the aiming solution angles for the selected weapon. The aircraft is commanded to fly the profile to achieve an aiming solution and to deploy the weapon, either the bombing maneuver for the bombing run or the aiming maneuver for the gun/rockets.

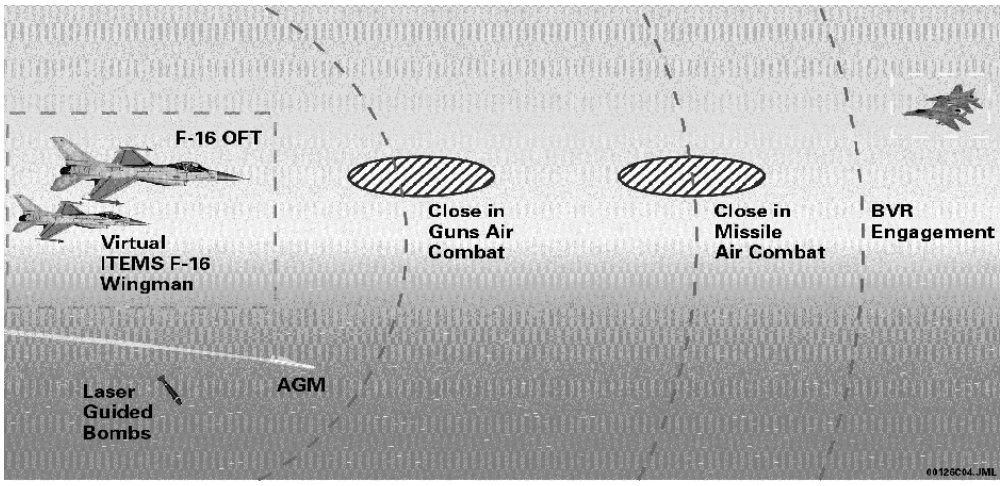

Figure 5. Elements of a Two-vs-Two Air-to-Air Engagement

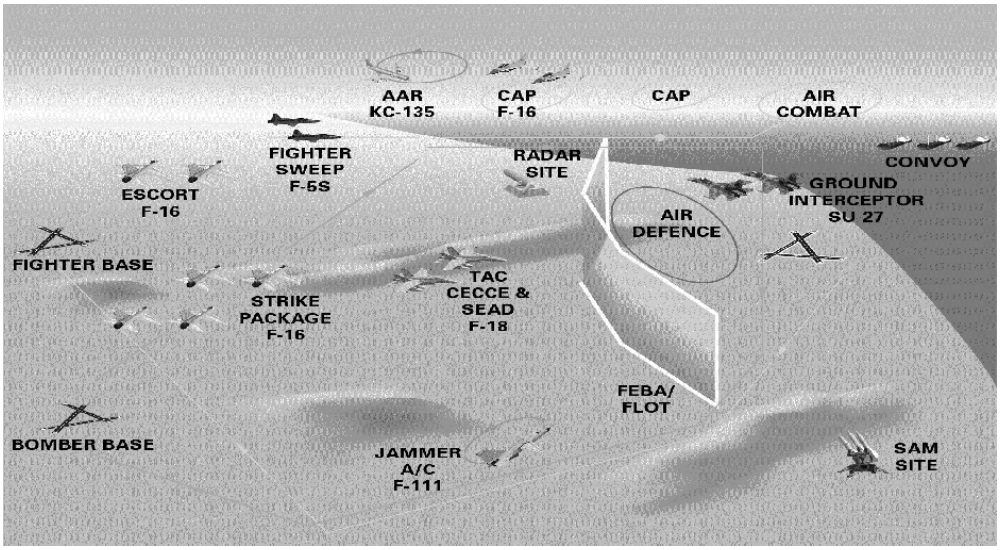

Figure 6. Elements of a Full-up Air Mission

12. Synthetic Environment Application Example 2

Added Complexity—Naval Synthetic Environments

Modeling Issues

Of primary importance is the integration of models within the tactical environment. This implies both correct tactical behavior of the model (i.e. Decision Making) and correct physical behavior.

Tactical behavior issues are generally very user specific. To this end, a solution that allows users to design their own tactics and tactical maneuvers is required. Design cycles with subject matter experts (SME) integrate the complexity and variability required by the user. A hybrid approach permits customized designs for each maneuver and also allows a rule-based definition to be used for any cursory tactics. This approach permits the system to remain flexible in case modified tactics are desired. The ability to inject (instructor) role-play into the system at any time also enhances the flexibility of the system.

Physical model behavior must be comparable to the behavior of systems on the manned simulator in order to avoid introducing dissonant effects that would lead to negative training. Such effects may include:

- Sensor (radar, sonar) differences based upon differing models used for CGF and simulator devices. This could result in detections by the CGF entities that are not consistent with those made by the trainee.
- Navigation differences based upon differing models used for CGF and simulator devices. This could result in CGF entities flying at different altitudes and speeds than the trainee.

To address these issues, models are built for *"closely coupled"* operation with the simulator. The models are either based upon the simulator model itself or are designed specifically to address the CGF model limitations. Within the simulation, model activation may be based upon proximity or some other appropriate measure.

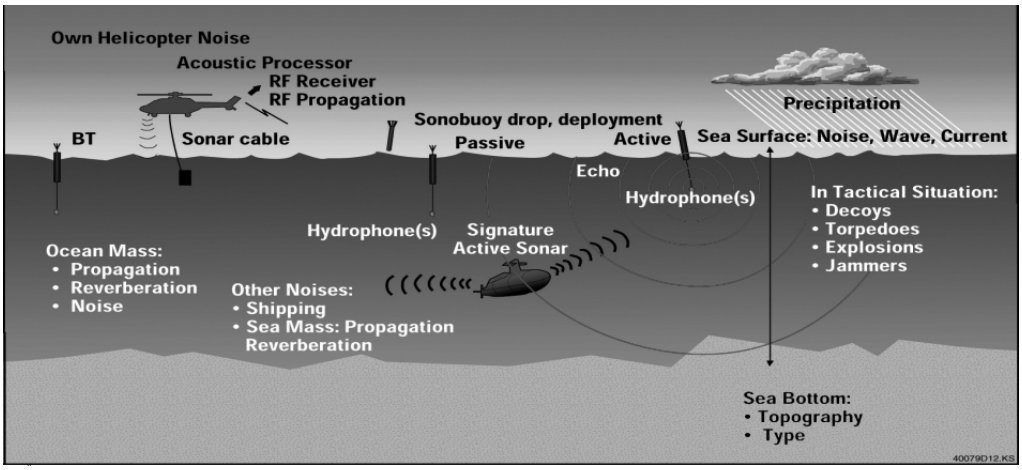

Figure 7. Acoustic Elements in a Naval SE.

13. Coordinated CGF Maneuvers

Coordinated CGF maneuvers represent a level of complexity higher than that for individual maneuvers. As such, they are both more complex in structure and present a more representative and challenging scenario for trainee interaction.

The coordination of entities within a requested tasking implies several units working together towards a single goal. Helicopters may be directed in a plan to search out a reported submarine. Ship groups may be engaged in a datum search or in tactical transit. Several unit types (e.g.: helicopter and ship) may combine in a coordinated tactic of submarine prosecution. A representation of the maneuver categories built could include:

- Helicopter group tactical searches:
  - Bearing and pattern searches for specified contacts.
- Helicopter and ship group tactical searches with target prosecution:
  - Bearing and sector searches for specified contacts. Target prosecution may be role-played or automated via entity rules.
- Helicopter and ship screening maneuvers:
  - Sector patrols with respect to a high value unit. Helicopter dip maneuvers.
- Missile attack reaction strategy:
  - Missile attack reactions for helicopters and ships

14. Coordinated Modeling Issues

While the same technical issues that were pertinent to individual maneuvers are still valid for the coordinated tactics, several new concerns are raised which are specific to the latter.

Primary to these is the method used for coordinating the maneuvers. This includes both sensor coordination and event coordination. Sensor coordination is important in order to ensure that all entities taking part in the maneuver share the same sensor picture. If they did not, synchronized approaches and targeting would not be possible. Similar physical sensor models attached to all CGF entities is one method of maintaining sensor correlation.

A second approach is via Link 11 and contact report message simulation ensuring that tactical contact data is distributed to all members of a communications network. In cases where the maneuver is entirely interface driven, user specified entries might be used.

Event or timing coordination must guarantee that all entities involved are aware of their role (what they must do and when). In order to ensure that entities can choreograph themselves as appropriate within the maneuver, specific events may be generated. These are either provided by the rule-based behavior model or by a state machine based representation of the maneuver progression.

Validation of entity capabilities either at the interface level or at entity definition further ensures that those entities nominated for a maneuver are capable of performing it.

Since the set of maneuvers discussed incorporates multiple entities, one or several simulators may participate by engaging any of the entity positions. In a four aircraft search deployment, for example, simulators may occupy any or all of the four positions, including lead. Additionally, the coordination methodology must support one or more aircraft positions that react independently, that do not provide all the event queues expected by the software and that may not respond as predicted. To support this level of flexibility and interoperability, the maneuver schedulers must have an open concept, as do the control interfaces.

15. Ongoing Developments: Adoption of the High Level Architecture

The High Level Architecture (HLA) is a communication protocol which offers a means to expand the level of fidelity associated with networked devices. The actual content of the information to be exchanged is left to the user(s) when creating the federation. This means that the level of fidelity associated with the data exchange need not be constrained by the standard or format. In future training systems, non time-critical systems such as RWR(defn?), radar and sonar are likely to be networked using HLA.

16. Ongoing Developments: Object Oriented Software Design

The emergence of software technologies such as Object Oriented design will affect what can be achieved by way of training capability as well as how future systems are built and used [1]. Object oriented (OO) design presents a completely new paradigm in building CGF environments. With it come such improvements as greater versatility in creating scenarios, scenario elements and doctrines. As an example, doctrines built in the OO environment are more easily capable of referencing any entity of interest in the scenario. Through plugins, new elements may be introduced into the scenario in a flexible manner without having to rely on pre-defined setups.

The concept of frameworks [4] is also being used to create a more modular open architecture, and one that allows components such as the knowledge-based system to be replaced without re-working the rest of the software. The effects of such advancement on the training arena are yet to be seen but are likely to imply increased realism of simulated systems and an increased coupling between the trainee man-in-the-loop and the electronic battlefield. The trend towards team-oriented training is also likely to continue and to increase in terms of its scope and its importance to the training community.

Some of the next generation software architectures are using the HLA concepts as an integral part of the design, for example OneSaf and STRIVE [5]. These new architectures readily lend themselves to HLA networking without the need for wrappers, and they maximize features of HLA such as time management. In addition, emerging standards such as the Real-time Platform Reference Federation Object Model (RPR-FOM) are likely to aid the simulation model development process by defining the model attributes that must be supported.

17. General Issues and Challenges

The elements of a human-in-the-loop simulation in SE must be integrated in such a fashion that they have, and act upon "knowledge" of each other:

- SE entities must have terrain knowledge and be able to reason with that knowledge
- SE entities must have knowledge about cultural features (e.g. Trees) and virtual players and be able to reason with that knowledge
- SE entities must display realistic performance (no 4000 knot fighter aero models or fully loaded infantrymen leaping tall buildings in a single bound) and must behave (make decisions and respond to the SE/virtual players) in a manner appropriate to their identity (e.g. combat ready fighter pilot, civilian non-combatant in the wrong place at the wrong time) and nominal level of expertise.

SE levels of control can be stacked hierarchically:

- Local command language—sensor/motor loops
- Doctrine/tactics command language—tactical behavior
- Doctrine/C3I command language—strategic behavior

Resolution and fidelity of SE model/database elements can vary substantially—e.g. in an acoustic model:

- Simple—only range may be modeled
- Moderate—include frequency, amplitude
- Complex—include reflections, Doppler effects, etc.

If physical and environmental models in SE systems are compose able, attributes may easily be added or enhanced.

Machine performance and behavior with a human-in-the-loop representation is easier to model and implemented more effectively than human entity representation per SE.

Time critical data must be processed separately from "informational" data.

Level Of Detail (LOD) of physical models and articulated elements must be integrated with knowledge of terrain and environmental conditions such that motion, particularly motion of human entities, is smooth and uninterrupted.

Visual database/texture resolution needs to be matched to entity/cultural feature Level Of Detail to reduce likelihood of visual artifacts such as "skating" and interrupted motion.

The adequacy of visual representation is less and less dependant upon graphics processing power as the technology continues to advance in accordance with Moore's Law.

The realism of visual models in a simulation (level of detail as well as motion fidelity) remains a challenge in realistic articulation of elements such as vehicle antennae and turrets, or, with human figures; posture, joint articulation, response to uneven ground—and is essentially dependant upon the quality and resolution of the physical models driving those entities.

A continuing challenge in both entity and physical models is the representation of multiple similar models (e.g. an infantry platoon), their lawful interactions, and the display of apparently independent behavior.

References

1. Gamma, Helm, Johnson, Vlissides: Design Patterns, Elements of Reusable Object-Oriented Software. Addison-Wesley Publishing Company.
2. European Computer Manufacturer's Association (ECMA): "Reference Model for Frameworks of Software Engineering Environments". TR/55, 2nd edition, December 1991.
3. IEEE P1516.1 Draft 1: High Level Architecture (HLA) - Federate Interface Specification. IEEE, April 1998.
4. Siksik, D., Beauchesne, G., Holmes, R., "The Integration of Distributed Interactive Simulation Protocol Within an Interactive Tactical Environment". Royal Aeronautical Society, London, 1994.
5. Howells, P.B., Charbonneau, M., Kwan, B., "Simulation of Fixed-Wing Air-to-Air Combat using ITEMS". AIAA Conference, New Orleans, August 1997.
6. Simon, E., Howells, P.B., "Experiences in Creating and Managing Networked Simulations". Networking in Simulation and Training Proceedings Royal Aeronautical Society, London, November 1998.
7. Howells, P.B., Simon, E., "On the Use of Frameworks for Real-Time Simulation Applications". Proceedings SIW Workshop Fall 1999.
8. Valade, S., Howells P.B., Simon, E., Gagnon, F., "On the Development of a Synthetic Tactical Real-Time Interactive Virtual Environment". Proceedings SIW Workshop Spring 2000. Spring 2000 SIW conference.
9. Enumeration and Bit Encoded Values for Use with Protocols for Distributed Interactive Simulation (DIS) Applications accompanying IEEE Stds 1278.1-1995 and 1278.1A-1998