

# DRAFT

## Trainer Architecture Evolution

**Richard H. James**  
Orion Development Group, Inc.

### 1. Introduction

#### 1.1 Purpose

The requirement for a robust data interchange capability to support networked simulation systems did not arrive overnight. Instead, this need for unambiguous, loss-less data interchange is due to the evolution of the computer system architectures used to support the training and simulation community.

The purpose of this paper is to examine this evolution of the architecture of training systems over the past two decades to show why there is now a need for an environmental data interchange mechanism such as SEDRIS. The architectural evolution of these computer systems has increased the demands on the environmental databases that support the training systems. Now, having a visually correct depiction of a geographic location is not enough. The data used to represent the simulated environment must also support other new features of training systems, such as computer generated forces and networked training systems that the new computer architectures allow us to employ. The need to interchange environmental data has evolved as well from trying to lower development costs through re-use to expanded training capability and interoperability.

#### 1.2 References

1. Transcript of John Norwood's Spring 1996 Presentation
2. Transcript of Farid Mamaghani's Spring 1996 Presentation
3. SEDRIS Management Plan - Section 2, 29 April 1997
4. Synthetic Environment Data Representation and Interchange Specification (SEDRIS) Rationale Document, Version Draft 0.2, 20 June 1996
5. Transcript of FAQuest held 23-24 July 1997
6. MIL-STD-1820, Generic Transformed Data Base Design Standard
7. MIL-STD-1821, Standard Simulator Data Base (SSDB) Interchange Format (SIF) Design Standard
8. DoD 5000.59-P, DMSO Modeling and Simulation (M&S) Master Plan, 17 October 1995

## 2. Training System Architectures

### 2.1 Stand-Alone Virtual Training System [3]

From the time Edward Link built his first “blue box” flight trainer up through the 1980’s, the architecture for a basic training system has been that of the stand-alone virtual trainer. Even in today’s networked training systems, the components of the network usually have the capability to also function as stand-alone training systems. The “stand-alone” architecture uses a dedicated computer or computer group to run a simulation program that creates a functional representation of the training audience’s tactical environment. Usually, this type of training system simulates the real world for small training audiences – a single person or a few crew-members. The simulated environment is usually only the interior of the trainee’s vehicle – a tank, aircraft, or ship. Initially, visualization of the natural environment or other vehicles external to the trainee’s vehicle was not provided, or it was provided using static scenes with 3-D physical models viewed through special mechanical/optical devices.

As computer image generation systems became available, visual scenes of the natural environment, as viewed “out-the-window”, were added to the stand-alone training system’s architecture. The ability to view a simulated scene of the battlespace greatly enhanced the training capability of these training systems. The first visual scene generation systems were added to flight trainers which usually had the trainee placed at a high altitude and moving rapidly through the scene. Therefore, only a low-level of fidelity was required for the representation of the natural environment which was primarily made up of the earth’s surface with land and water body masses. The only high fidelity modeling of ground features was done near airports and runway areas.

In the early stand-alone training systems that included an image generator, the natural environment representation had virtually no interaction with the trainee’s own-vehicle as well as no direct interaction with any other simulated entities in the training exercise. In fact, due to processing limitations in the computer systems, there were a minimal number of these simulated entities (*i.e.*, other aircraft including enemy aircraft and munitions). The simulated entities were usually under the direct control of the training system’s operators and the instructor. Again, due to limitations of the computer systems, there was no control of the simulated entities by the computer. In addition, other than the trainee’s visual sensors (*i.e.*, their eyes), there was minimal interaction between any simulated sensors and the simulated natural environment.

The image generator systems were driven by specially constructed databases containing the representational data necessary to create a visual scene of the natural features of the battlespace. These databases were optimized for visual scene generation since that was their only use. As the image generation technology improved, the capability to depict a specific geographic area with sufficient fidelity that the trainee could identify his “location” emerged. However, the effort to produce these environmental databases was not trivial especially when the data represented a specific geographic location in the world. If a stand-alone training system had a visual representation of Central Europe or Southeast Asia, it was quickly recognized that substantial developmental time and money could be saved by re-using that environmental database in

## DRAFT

another training system that also needed a visual representation of the same geographic areas. Unfortunately, environmental databases were custom developed to support a specific, proprietary image generator system. Although the databases contained representational data of the same geographic location, the databases could not be shared between different image generation systems because of differing data formats and implementations.

Recognizing the need to minimize the costs of formatting specific geographic data for inclusion in environmental databases, the Air Force began Project 2851. This Project was an attempt to establish a means of interchanging the environmental data between different training system's databases. The results of Project 2851 were MIL-STD-1820, Generic Transformed Data Base Design Standard, and MIL-STD-1821, Standard Simulator Data Base (SSDB) Interchange Format (SIF) Design Standard. With these two standards and their associated standard data formats, it was felt that the representational data for a given geographical location could be shared between two different training systems even though the systems had different image generators. This data sharing would greatly reduce the costs of developing environmental databases. The SSDB was established as the central repository for the validated simulator databases for the DoD training simulation community. The SIF serves as the input/output vehicle for sharing digital simulator databases via the SSDB. The primary focus of Project 2851 was to have a database re-use capability for environmental databases that supported stand-alone training systems, primarily those used for high-altitude flight training.

### **2.2 Mobility Training System [3]**

The next step in the evolution of training system architectures is perhaps not a full step "architecturally" from the stand-alone systems. In fact, mobility training systems have the same computer architectures as stand-alone systems. They still have the trainees physically in the simulator with out-the-window views of the natural environment. The major "architectural" difference with the mobility training systems is the demand placed on the data representation of the battlespace.

In a mobility-oriented training system, the mobility model of the trainee's own-vehicle or own-ship is directly affected by the data representing the natural environment. A simulation model of a tank driving over a terrain is influenced by the simulation of the terrain. The data attributes associated with the terrain model such as slope, soil composition, and soil condition (*i.e.*, wet or dry) are used by the mobility model so the trainee experiences different responses from the simulated vehicle. A simulated ship on the surface of the ocean, or a submarine below the surface, is affected by the data attributes of the ocean model. Sea-state and currents cause different handling conditions for the simulated vessel. Even a simulated aircraft, especially one designed for low altitude flying, is affected by both the terrain it must avoid as well as the atmospheric conditions (*e.g.*, wind, rain, clouds) that impact it. The critical impact of this training system's architecture on the environmental databases is that these data attributes, needed by the mobility models, are not the type of attributes included in the databases for the visual representation of the environment. New data types and attributes are needed in the environmental database along with the new requirement that this data be correlated with the

## DRAFT

visual data. If a trainee “sees” a muddy road, then the vehicle must respond as if it is being driven through the mud.

In addition to the mobility models of these training systems, the simulated sensors employed by the trainees can also be “affected” by the simulated environment. This is especially true of naval sonar trainers as well as air, ship, and ground trainers using infrared, thermal, ~~and~~ night-vision, ~~sensors as well as~~ radar and electronic warfare sensors. All of these new requirements on the training systems are not the result of new tactical systems – they are new because the simulation capability of the computer/software systems have improved so this increased degree of realism with these sensors can now be created in the training systems. But, to have the environmental affects on the simulated sensors, even more new data types with new attributes must now be included in the environmental databases and correlated with the other data representations.

One other aspect of the “architecture” of training systems that has also changed due to greater computational capacity is the increase in the number of simulated entities (again, entities meaning other simulated vehicles, weapons, or combatants) in the battlespace. Larger, more complex battles can now be simulated. Team trainers, where all members of a tactical team are involved simultaneously in the training exercise, are now being employed. The number of simulated entities has become so great that the instructors and operators of the training system can not control them. These entities must now be “intelligent” and interact with the simulated environment and the other entities (including the trainee’s own-vehicle) according to computerized behavioral models. Architecturally, the training systems’ computational capability can handle the behavioral modeling needed for these computer generated forces (CGF). However, new data types and attributes are required in the environmental database to support this behavioral modeling. What types of data, and in what form, are needed to determine which simulated roads a computer generated vehicle should use to get to a desired location in the simulated battlespace? The environmental database must now support these types of “reasoning” models.

One can see that the data that models the natural environment must now do more than just drive an image generator system. The mobility models of the various vehicle entities encountered in the battlespace must be supported by the data representing the environment. The models of the sensor systems must use environmental data to appear realistic to the training audience. The computer generated entities need to react realistically with the environment. The issue of correlation, meaning that the environmental affects need to be applied in real-time simultaneously to all aspects of the simulation, has become a major factor in the fidelity of the training system. We are now dealing with more than just the natural environment. The new term used to include the totality of the environmental dataset needed to support a mobility training system employing sensor simulations and CGF is *synthetic environment*.

Yet the database generated for the synthetic environment is still primarily focused on the image generator. The mobility model interactions with the environmental data as well as those of the computer generated entities are dealt with on a training system by training system basis. These are still stand-alone trainers with unique requirements and unique implementations. How

## DRAFT

this trainer architecture is used to deal with mobility and multiple simulated entities is unique to each training system. The data exchange between environmental databases as defined by SIF does not address these new data types and attribute demands of the synthetic environment databases.

### 2.3 SIMNET [3]

With the reduction of the military budgets that began in the late 1980's, the ability to gather large numbers of a training audience at one location and conduct multi-echelon training was severely impacted. Therefore, the concept was developed for wide area networking of the existing stand-alone training systems. These networks of training systems would involve the multi-echelon training audience but from their individual home bases using their local stand-alone training systems. Travel to a central training site would be minimized. The research project that developed this networking capability was SIMNET. Networked training systems is the next step of the architectural evolution of training systems.

Initially, SIMNET focused on integrated networks of homogenous training systems. A tank trainer at Ft Hood would be networked with the same type of tank trainer at Ft. Knox. Although the training audience was geographically dispersed, the synthetic environment used in the common training exercise would be identical since the trainers involved were exact copies – each had the same, exact environmental database. Therefore, the trainees could train on the same problem in the same simulated battlespace. This is called *interoperability* – two training systems interoperating to present a single training exercise in the same battlespace to a geographically dispersed training audience.

This new architecture for training systems created new technology problems from those encountered in the stand-alone systems. Now, computer generated entities viewed in the scene in one training system could also represent either a vehicle manned by trainees in the other training system or a computer generated entity controlled by the other training system. The local stand-alone training system was not in “control” of all the entities in the battlespace. When a vehicle controlled by one trainer makes a turn to the left (in other words, changed states), how does the second trainer learn about this state change so it can update its visual scene to show the left turn?

This control problem was solved by passing appropriate data between the training systems that described the entity state changes occurring during the execution of the training problem. Each training system could use this state data to update its local representation of the battlespace. Since the training systems were exactly the same, it was a straightforward task to identify a dataset to interchange for describing, and responding to, state changes. Prior to the start of the exercise, the operators of the networked training systems would establish the network interfaces, and agree on the specific synthetic environment that would be used for the exercise. Since both training systems used copies of the same database, no synthetic environment data would have to be exchanged prior to, or during the running of, the training

## DRAFT

exercise. The “state” of the synthetic environment did not change during run-time of the exercise.

Naturally, the idea of interoperating training systems had such universal appeal that the concept was extended to heterogeneous (*i.e.*, built by different contractors with different image generator systems) training systems. Unfortunately, this created new problems – not only to the run-time data interchange of state changes – but, due to the different environmental databases used in each training system. Although the geographical area to be used for a training exercise was to be exactly the same, the run-time implementations of the two databases in the individual training system was different. These differences manifested themselves when viewing the scenes of the synthetic environment in each trainer – they didn’t look exactly the same. For example, the terrain features were not the same. A depression in the terrain depicted in one training system was not in the other. Therefore, if a tank was “driven” into this depression to hide, trainees in the other trainer would still “see” this tank (remember they didn’t have the depression) – and would shoot at the first tank. In another example, the slope on a portion of the terrain may be determined by one training system to be too great to allow mobility, while in the other training system the modeling was not as precise, the slope was not too large, and it allowed vehicles to “drive” across this sloped portion of the terrain.

These examples describe what is known as the *fair fight* issue. For two training systems to interoperate together on the same training problem, they must have a common, *correlated* view of the synthetic environment. There are other aspects of the fair fight issue, such as the difference in the fidelity of the modeling of the computer generated entities, but the sharing of the environmental databases between training systems is one of the major issues that must be resolved for interoperability. The evolution of training systems’ architectures has pushed the synthetic database community again. The synthetic environment data sharing issue is now more than just re-use – although that cost saving benefit is still viable.

### 2.4 CCTT [3]

In the early 1990’s, STRICOM initiated the Close Combat Tactical Trainer (CCTT) Project, the first of the Combined Arms Tactical Trainer (CATT) Family of training systems. CCTT was designed to correct several criticisms of the SIMNET systems – in particular, to have a high-fidelity representation of the environment, especially the terrain. CCTT would also have a greatly increased number of CGF entities. The CGF entities would be controlled by advanced behavioral and reasoning models that would realistically simulate each entity’s behavior in combat including its interaction with the environment. Finally, CCTT was developed to interoperate with other trainers – other CCTT systems, other training systems of the CATT Family, and other non-similar, heterogeneous training systems.

CCTT would use the new Distributed Interactive Simulation (DIS) Protocol Standard, IEEE STD-1279, for all data communications both internally to the trainer system as well as for interoperability with other trainers. The DIS protocol was developed to support the new training systems architecture introduced by SIMNET – wide-area distributed heterogeneous training

## DRAFT

systems. Since the internal architecture of CCTT was using a network of computers to control the simulation exercise, DIS was also used to pass data between the internal computer nodes.

These new requirements further increased the demand on the data representations in the environmental database. The visual scenes required a high level of fidelity, yet had a limited number of polygons to “draw” the scene. The CGF models needed different data types and attributes from the environmental database to support their decision-making. Two-dimensional views of the battlespace with map symbology and labels were needed for both paper map generation as well as for the instructor displays. The high-fidelity mobility modeling of the vehicle simulations also added to the data demands on the synthetic environment database. On top of all of these different data representational demands on the synthetic environment database was the need to have a correlated “view” of the battlespace during run-time. The tank model needed to know that it couldn’t go up the hill due to its high slope so the trainee could learn that the hill he saw out his viewport was too steep to drive on. The CGF model of dismounted infantry needed to know that the 3-D building model has doors and windows and that “they” may go into the building for concealment. During run-time, each of these CCTT subsystems needed a different representation of the data features from the synthetic environment database.

Since the SIF standard did not include representational data types beyond those needed for visual scene re-use, an expanded SIF, called SIF++, was created for use on the CCTT project. A single “source” database was created for the synthetic environment used in CCTT. This source database attempted to contain all of the data representations needed by all of the CCTT subsystems. The visual database was extracted from the source database and then compiled for run-time use by the image generators. SIF++ was used to extract data from the source database for the other subsystems of CCTT, such as the CGF and the Plan View Displays, that needed environmental data.

Each set of data extracted from the source database was compiled into a unique run-time database, structured to support a specific CCTT subsystem. These unique databases were necessary because each subsystem needed the environmental data in a different representational format. The data that was extracted from the original source database still had to be modified for optimal use in each of the run-time databases. The extraction by the SIF++ mechanism did not provide the data in the form needed by each subsystem. This created the problem of how to do the data extraction and transformations while preserving the correlation between the various representations of the environmental data.

Although the CCTT program has turned out successfully, it was recognized that the SIF++ extension to Project 2851’s SIF Standard was still not sufficient for exchanging environmental data for interoperability – even when the “interoperability” was just between internal subsystems of a network architectural training system. Continued modification of the representational data created a correlation problem. This experience provided a strong indication of the difficulty that would be faced when interoperability with other training systems was attempted. However, much was learned about what is needed to provide the representations of environmental data for use by CGF and the other internal users of environmental data while also providing a dataset for the visual scene. The use of separate run-time databases generated from a source database could

## DRAFT

be a solution to the problem faced in interoperability – using multiple sets of data representing the synthetic environment on different systems.

The CCTT system produced the next evolution in training system architecture – an internally networked system using different environmental databases at run-time. But, the data requirements on the environmental database have now grown to the point where a true data interchange mechanism is needed to support both re-use and interoperability, while satisfying the data representation needs for more subsystems than just the image generator. The SIF++ extensions used on CCTT were not the solution to this problem, but the experience did show the way to the solution.