INITIALIZATION OF UNIX BASED SIMULATION EXERCISES FROM THE PERSONAL COMPUTER

Jeffrey B. Abbott
AcuSoft Inc.
12249 Science Drive Suite 160
Orlando, FL 32826
407-658-9888

Mr. Jeff Abbott is a project manager for AcuSoft Inc. He holds a Masters Degree from the University of Central Florida in Electrical Engineering and Communications Systems. He has 14 years experience in the development and management of software systems for government contracts. He is currently working on the CCTT Exercise Initialization Tool and performing interoperability research for simulation in the areas of interoperability test/compliance, and composable/standardized simulation architectures.

Dr. Mona Crissey
ARL-HRED-STRICOM
12350 Research Parkway
Orlando, FL 32826-3276
407-384-3639

Dr. Mona J. Crissey, ARL-HRED-STRICOM, is Project Director, Training Exercise Development System (TREDS) for Program Manager Combined Arms Tactical Trainer (PM CATT) at STRICOM. She holds degrees in Education from University of Alabama (Ed.D), University of Kentucky (MA), and SUNY Cortland (BS). She has 16 years of program management and database development experience with government and industry. She is currently the government lead for development of the training environment for the Aviation Combined Tactical Trainer-Aviation Reconfigurable Manned Simulator (AVCATT-A) to include After Action Review (AAR) and Mission Control.

ABSTRACT: *To effectively train in simulation environments, all aspects of mission planning must be considered. This planning can occur in locations far from the simulation site, and can involve any of the following: development of the scenarios expected to be executed, preparation and generation of support products such as the operations orders, maps with overlays, execution matrices, and administrative orders, and initialization data parameters for both the equipment to be used and the simulated battlefield. Today's military planners are becoming increasingly computer literate, however, many trainers do not use simulations often enough to become familiar with the specific exercise planning interfaces used in their simulation environments. Although, simulation has moved much closer to the desktop, seamless initialization will require easier migration of common desktop tool produced products into the simulation. Familiarity of user interfaces for simulation planning and initialization tools can be achieved through the integration of desktop commercial off the shelf (COTS) products already in common use. Most exercise plans originate in COTS products such as Microsoft Office. These plans are then prepared for application to specific simulation environments. The process of exercise preparation is primarily a task of transposing the mission, overlays, and execution matrices from desktop formats into formats compatible with specific simulation environments and can become a time consuming process. Desktop software is now sophisticated and powerful enough to automate this translation of exercise plan information (documents) into simulation formats for exercise initialization. This process involves the transfer and integration of a simulation environment's object model into common desktop tools. The object model must be integrated with the planning tools so that through the course of exercise development, the plan may be captured in a format consistent with scenario generation. Initialization of Close Combat Tactical Trainer (CCTT) exercise units, obstacles, and control measures from PowerPoint will be used as a practical example of this functionality. In particular, the paper will show how COTS products meet user requirements for an automated initialization tool and provide a uniform view of the training and simulation domains for both users and developers.*

INITIALIZATION OF UNIX BASED SIMULATION EXERCISES FROM THE PERSONAL COMPUTER

Jeffrey B. Abbott
AcuSoft Inc.
12249 Science Drive Suite 160
Orlando, FL 32826
407-658-9888

Dr. Mona Crissey
ARL-HRED-STRICOM
12350 Research Parkway
Orlando, FL 32826-3276
407-384-3639

BACKGROUND

The following paragraphs present past development efforts that provide the keys to integrating familiar Windows desktop applications running on personal computers with simulation applications running on RISC based processors under UNIX or UNIX like (AIX) operating systems for the purpose of exercise initialization.

Early in the development of the CCTT system, a need was identified for a comprehensive training program that would provide the lesson plan level information required for units to successfully conduct training, take advantage of CCTT features and capabilities, and avoid the problems of negative training. As part of that effort, an automated planning tool that would enable commanders and training developers at home station to generate training exercise materials as well as manage exercise data at the collective task level was developed. The Training Exercise Development System (TREDS) provided a library of Mission Training Plans (MTPs), a map library, a scenario library of CCTT scenarios, and a variety of modules designed to help automate common processes such as generation of After Action Review slides and Training and Evaluation Outline (T&EO) task lists. Recently, another tool, the Commanders Integrated Training Tool (CITT) has been developed for CCTT that will extend these capabilities further.

Another tool developed for use by commanders was the Unit Off Site Planner (UOSP). The purpose of this tool was to enable scenario developers to completely plan the simulation initialization details on home station PCs, transfer that data to diskette, and then provide the CCTT site manager with a completed scenario. He, in turn, would input the details concerning the planned exercise into the CCTT Master Control Console (MCC) and automatically initialize the simulation battlefield according to the unit's plan.

During a BAA based study on CCTT Interoperability with an Aviation manned module simulation, the need arose to enable the definition of CCTT exercises in a reusable ASCII format. In the course of the BAA, a PowerPoint tool for laying out battlefield graphics was extended to support the UOSP format in CCTT. This extension enabled PowerPoint to export CCTT exercise data from any Windows based PC running Microsoft Office 97.

With development of the Synthetic Environment Data Representation Interface Specification (SEDRIS), an architecture was created to allow visual system terrain and feature data to be shared across hardware and software platforms. The key to such standardization of data reuse was the development of a SEDRIS API that is cross platform and cross operating system capable. As a result, any application that supports the API can reuse SEDRIS data in a loss-less fashion. The single API approach provides a common interpretation of SEDRIS data enabling interoperability across visual system platforms.

All of the above tools and research have provided parts of the initialization package. There are still some problems to be found, the main one being that of translating data produced on the PC into usable data for the simulation which, in the case of CCTT, is hosted on UNIX. This directly usable data will provide another major benefit as well, that of saving considerable time now required to be spent by CLS staff in development of exercise files.

UNIFORMITY

The first step in interoperability between the exercise planning process and simulation is to provide a uniform

view of the simulation domain to both the simulation developer and user (training developer). Much of the information contained in training materials must be used to initialize simulation-based exercises. The problem is that little or no data from training materials can be transitioned directly to an exercise's initialization. To provide a uniform view of the simulation domain, information requirements of the training domain must be mapped to the data requirements of the simulation domain.

If a uniform view of the training and simulation domains is to be obtained, certain assumptions must be made and tested. The most important assumption is "The basic training process does not vary by training environment." This statement would suggest that whether in the live, virtual, or constructive environment, these modes of simulation based training are equivalent. If we accept this hypothesis, then the simulation domain falls within the training domain and we can conclude that the simulation process does not require the user to perform any tasks beyond those required for live training. However, since simulation involves automation, some of the difficult tasks in the training process ought to be simplified, and some of the tedious tasks ought to be automated. Statements as bold as these are not likely to be accepted at face value. Let us examine the basic steps in the training process to see how well simulation maps to them.

Simulation and Training Exercise Steps

Four phases are common to any training exercise. They are Plan, Prepare, Execute, and Assess. We will restrict the scope of this paper to the steps of Plan, Prepare and Execute. The Evaluation or Assessment step that comprises the After Action Review process will not be discussed, although it is clearly linked to the other three.

Planning is the first step in the training process. It involves the development of exercise objectives and the conditions under which the exercise is to be performed. Conditions may be defined as the Mission, Enemy, Terrain, Troops, and Time (METT-T) constraints for the exercise. Each mission impacts on the behaviors and tasks to be simulated and/or performed in an exercise. Enemy and Troops identify the units to be trained as well as simulated. Terrain identifies the terrain database (battlefield play box) and impacts on the locations of units and control measures. Time available identifies constraints on schedule, duration, and exercise synchronization.

Preparation for an exercise generally involves scheduling for, and acquiring exercise assets. This step maps well to simulation. It is the preparation step in live training that can be the most difficult. Here, simulation greatly simplifies the difficult tasks of obtaining exercise assets such as equipment, fuel, ammunition, etc. Generally, all that needs to be done is to schedule the exercise with the simulation site, prepare for food, lodging, and travel, and given an exercise initialization file, the simulation will generate (automate) the rest.

Execution of a simulation-based exercise involves more than just performance of the tasks to be trained. Execution involves appropriate stimulation of the training environment in circumstances designed to train soldiers to perform appropriate tasks to standard and to stress the training audience's abilities to fulfill the exercise mission. In simulation, the task of stimulating the training environment primarily falls to computer-generated forces (CGF), semi-automated forces (SAF) operators, unit commanders, and other role players and training participants.

Evaluation and assessment are completed during the After Action Review for an exercise. This step involves the collection, review, summary, and analysis of exercise data that was created or collected from the other steps.

Of these four steps in the training process, the planning step benefits the least from simulation. However the information and data derived from this phase must be used to execute a simulation exercise. If the exercise plan captured in training materials is to map into the simulation domain, the plan must not only present information, but it must also capture and deliver the data required by a simulation environment to implement the exercise.

Data are used cognitively by the trainer to develop information used in the definition of a training exercise. If these data are not captured during the development of training materials, they are lost. Remember our assumption: "The basic training process does not vary by training environment." If training developers are to use the same process to implement a plan for simulation as they would use for live training, then the tools they use in real exercise planning should be extended to capture that data required to initialize simulation. It is important to note that if we replace the trainer's tools with those developed for a specific simulation environment, then we have violated our assumption. This makes sense from a human factors perspective as well. People generally use those tools they are familiar with, even if it means more work. Thus, because it is unlikely that a simulation user will train in a particular simulation environment often

enough to become familiar with its planning tools, it is most likely those tools will not be used. This is not a question of human-machine interface. No matter how well an interface is defined and implemented, familiarity ultimately comes from experience. The challenge, then, is to capture simulation relevant data in the user's everyday tools.

APPROACH AND IMPLEMENTATION

To enable the capture of simulation data during the planning process, three tasks must be accomplished:

- Port the simulation environment data requirements to the user's desktop.
- Link the ported data to the trainer's planning tools and environment.
- Motivate the user to capture the data required by simulation.

As a practical example of this approach, CCTT will be used for the simulation environment, and PowerPoint will be extended to perform the planning that is currently done on the CCTT Plan View Display (PVD). To achieve integration of CCTT with PowerPoint, heterogeneous software and hardware issues need to be resolved. A porting strategy that allows CCTT to share data with PowerPoint across operating systems (AIX and Windows) and hardware platforms (Motorola PowerPC and Intel Pentium platforms) will have to be developed.

Porting Simulation Environment Data

Let us return to the earlier examination of an exercise's METT-T conditions as they relate to simulation. The Mission, Enemy, and Troops identified for exercises are constrained by the types of units supported in the target simulation environment. The terrain is constrained by the terrain databases available in a simulation environment. Unit type data alone are not sufficient to support initialization of units in simulation. Other data, to include unit designations, formations, behaviors, and specific manned module definitions, are needed. Likewise, identification of the terrain database to be used is not sufficient. In order to place units, control measures, and entities in their proper positions for the exercise, correlated digital maps are required. Digital maps, whether rendered from terrain databases or scanned, must be incorporated into the graphics package(s) used by the training developer to enable the capture of unit, entity, and control measure positions and orientations. In an ideal situation, all data required to plan exercises for a simulation would be ported to the PC desktop in the simulation environment's native format. When the data do not exist, are incomplete, or incompatible, formats will have to be developed to complete this task. In the case of CCTT, PVD databases exist that may be read on the PC to render correlated digital maps. ASCII reader files and tables also exist which document the supported units, unit placement parameters, unit behavior parameters, supported entities, and entity configurations. These files may be easily read (parsed) on any hardware platform.

Terrain support is provided by the virtual terrain databases (VTDB). These databases are used by SAF for terrain awareness, and map rendering on the PVD. In CCTT the VTDB PVD database is used to render terrain features (roads, fields, trees, etc), contour lines, and fixed selectable features (towers, bridges, etc). The VTDB MrTDB represents the terrain skin and terrain features. The VTDB MrsTDB represents the routing database used to generate routes for SAF entities to travel along. The CCTT PVD databases were ported to the personal computer in their original format to enable rendering of PVD maps under Windows. Units in CCTT (enemy, and friendly troops) are defined in reader files and tables (Table 1).

Table 1. CCTT Reader Files and Tables

`blufor_pure_units.rdr` Identifies the BLUFOR units in terms of their type, aggregate type, echelon, organization, equipment and life form makeup.
`opfor_pure_units.rdr` Identifies the OPFOR units in terms of their type, aggregate type, echelon, organization, equipment and life form makeup
`task_organization_values` Valid unit designations (example: 1-7CAV)
`mcc_formations.rdr` Identifies formation and spacing by unit echelon
`unit_placement_data.rdr` Identifies formation order of subordinate units by aggregate unit type
`vehicle_class_data.rdr` Groups entities by vehicle class.
`cgu_resource_data_file.rdr` Identifies fuel and munition loads, cargo, and repair capabilities for supported entities.
`aircraft_hardpoint_data_file.rdr` Identifies weapon system and ammunition loads for aircraft.
`uosp_soldiers_platforms.rdr` Identifies weapon systems by type of dismounted infantry.
`cis_data-[`*`agg_unit_kind`*`].rdr` Identifies the behaviors and associated formations

for each kind of aggregate unit supported in CCTT.

Initialization of unit behaviors requires each behavior be linked to specific units and appropriate control measures (routes, checkpoints etc.). Since the relationships of behaviors and control measures are not documented outside of CCTT source code, formats that capture those relationships were created and populated in reader files.

Linking Ported Data to Desktop Planning Tools

Linking the ported data to the exercise planning tools on the PC desktop involves four primary steps. First, object classes need to be developed to support loading of the ported data. These object classes must describe the properties, makeup, and behaviors of all simulation objects that may be initialized in an exercise. A general list of such objects includes, but is not necessarily limited to, units, vehicles, control measures, and environmental features such as obstacles and minefields. The following table depicts counts of these classes

Table 2. CCTT Object Counts

CCTT Object Classes	Counts
Unit Classes	41
Entity Types	96
Control Measures/Obstacles/Minefields	64

The second step is to instantiate a simulation environment's objects into the desktop exercise planning applications, making them available to the application itself. The third step involves developing software to enable the trainer's desktop applications to create and manipulate simulation objects as required in support of specific exercises. This software can be written in an application's native macro language, and/or linked into the macro language through dynamic link libraries (DLL), dynamic data exchange (DDE), or other methods. Applications that use Visual Basic, such as those found in Microsoft Office, are ideally suited for this task. The fourth step in the linking process is to provide a facility for saving the exercise plan in a format consistent with the targeted simulation environment.

Definition of Object Classes

In our example, Visual Basic for Applications (VBA) was used to define the object classes contained in the reader files. The reader files define CCTT classes to include supported units, entities, vehicles, munitions, control measures, weapons, orders (combat instruction sets), and others. The VBA classes were designed to mirror the relationships, structure, and format of the reader files.

Instantiation of Objects in PowerPoint

A C++ dynamic link library (DLL) was used to instantiate the Visual Basic object classes. The DLL provides support for reading, writing, querying, relating, and manipulating reader files. This DLL represents a critical piece of the overall application. The specific details of which will become clear in the following sections.

Integration with PowerPoint

CCTT objects as defined in Visual Basic were integrated with PowerPoint's own application interface through the development of Visual Basic code and reader files (Table 2). Digitized (PVD) maps were integrated through the use of a second C++ DLL in combination with a Visual Basic wizard-like interface.

Visual Basic code was written to render unit, and overlay symbol shapes from custom toolbars. Each PowerPoint shape is tagged with specific names as defined in the reader files. These reader files document the relationships between the PowerPoint shapes and CCTT Objects for Units and Control Measures. Following sections present additional detail on this subject.

Table 3. PowerPoint Integration Reader Files

`ceit_unit_data.rdr` Identifies the relationships between the unit types from blufor_pure_units.rdr/opfor_pure_units.rdr and unit shapes in PowerPoint. This file integrates CCTT unit definitions with PowerPoint unit shapes.
`ceit_overlay_types.rdr` Identifies the relationships between the overlay symbol types supported in CCTT and the overlay symbol shapes in PowerPoint. This file integrates CCTT overlay symbols with PowerPoint overlay symbol shapes.
`cis_detail-[agg_unit_type].rdr` Identifies the relationships between unit behaviors and the control measures needed to execute the behaviors in CCTT. This file integrates the order-specific parameters from CCTT with PowerPoint, enabling PowerPoint to deal with each behavior's unique parameters (routes, checkpoints, phase lines, etc.).

PVD map support was integrated into PowerPoint using a C++ DLL to render the user specified *"play box"* as a bitmap from the VTDB PVD databases. A wizard in Visual Basic leads the user through the process of identifying this "*play box*". The DLL is then used to

render the specified area as a sixteen-color bitmap. The bitmap is inserted into a PowerPoint presentation as part of the slide master (background). The PowerPoint presentation coordinates are correlated to 10 digit grid locations. The parameters defining the correlation from PowerPoint coordinates to MGRS coordinates are embedded within the presentation as properties. Next, the presentation is saved as a template under the "Template" directory of Microsoft Office. In this fashion, each time a user selects the "New" option under the "File" menu the template is made available for selection in the standard "New Presentation" dialog (Figure 1).

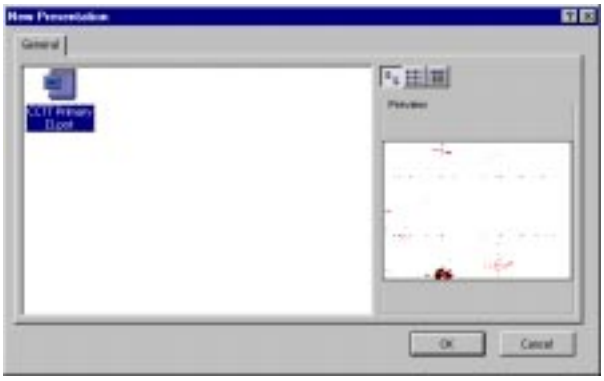

Figure 1. PowerPoint New Presentation Dialog

Once a template has been selected, the map may be cropped internal to PowerPoint to fit the requirements of the user. The map may be cropped prior to, during, and after specifying units and control measures. Once cropped, PowerPoint will display only those exercise objects that fall within the displayed area. Those objects that do not fall entirely within the displayed area become read only until the map is restored to full size.

Save in CCTT Format

Now that objects from CCTT, and the terrain have been integrated into the PowerPoint interface, the exercise defined by these objects must be exported in a format compliant with CCTT. Because both CCTT and PowerPoint need to be able to read and write the exercise file, it is very important to guarantee each interprets the content of the file in the same fashion. This requires both systems use the same API to perform the read and write tasks. The reader file DLL provides the solution. Reader file template definitions were included in the DLL to force compliance of the exercise file to a specific structure. The template supports validation of each field's values against the contents of the ported CCTT reader files. Because only one API exists, code maintenance is greatly simplified while guaranteeing a common interpretation of the exercise file.

Let us review the overall flow of data for the CCTT exercise initialization tool (CEIT) (Figure 2). The CCTT objects are defined internal to PowerPoint as Visual Basic classes. The classes were instantiated as objects using the Reader File DLL to load, query, and manipulate the ported CCTT reader files. Custom reader files were developed to integrate the objects under PowerPoint's own user interface using Visual Basic. A PVD DLL was developed to render correlated terrain maps for display in PowerPoint. The reader file DLL was again used to generate a CCTT compatible exercise file in reader file format under Windows. The same reader file DLL is then used by CCTT's MCC to import the exercise file under AIX.

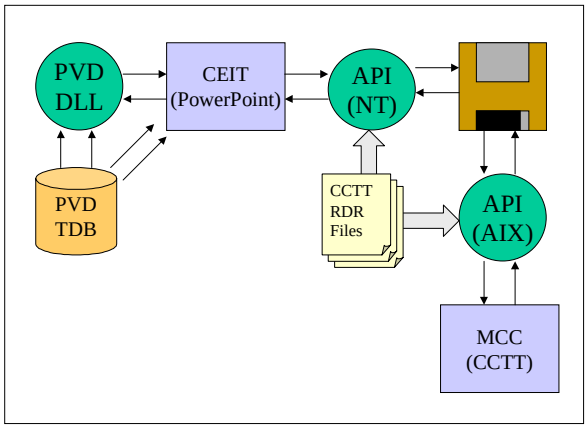

Figure 2 CEIT Data Flow

The common API approach taken for the API development is not new. SEDRIS used the same approach for development of the SEDRIS master database.

Motivating the User to Capture Simulation Exercise Data

Porting the data from a simulation into a desktop application and enabling the application to act upon those data to initialize a simulation exercise is a good start. However, if training developers are not sufficiently motivated to use the tools that have been integrated into their desktop, or if by accessing the new capabilities the task of training development becomes more difficult or tedious, the developers are likely to return to techniques that are more familiar to them. In order to motivate the training developer to use the simulation planning tools, those tools need to simplify difficult tasks, automate tedious tasks, save development time and be convenient to use. In our

example, the macro procedures automate the tedious tasks of delineating overlay symbology by converting simple shapes such as lines, rectangles, and ovals into units, minefields, bridges, axis of advance, front lines, fortifications, etc. The correlated digital maps that are stored within PowerPoint templates simplify the difficult task of determining the grid positions of units, entities and control measures. By integrating a simulation's planning functionality into the desktop, we have already made simulation planning more convenient. However, this may not be sufficient. In our example, seven toolbars were created to provide access to CCTT simulation planning functionality. Seven toolbars would certainly become cumbersome and clutter the screen. Each time one is to be accessed, the user must click on the "View" menu, select the "Toolbars" option and finally click on the toolbar of choice. To make this more convenient, an eighth toolbar was created. This toolbar consolidates the PowerPoint drawing tools needed to produce the simple shapes used by the macro procedures. This toolbar also makes access to the planning toolbars more convenient by incorporating shortcuts to them.

WALK-THROUGH EXAMPLE

A walk-through example of creating an overlay in PowerPoint for a CCTT exercise will reinforce the effectiveness of integrating simulation into the training environment. Each step will be discussed in terms of the impact that integration of CCTT data and functionality has had on the planning process. First, a new presentation must be created. Selecting the "New…" option under the "File" menu will allow a user to select terrain, based on the digitized maps available. In this step, the terrain is automatically identified and displayed in PowerPoint (Figure 3).

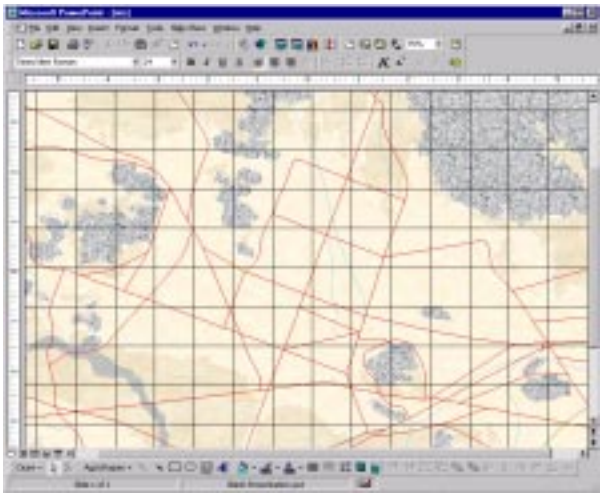

Figure 3. PVD Map – Primary 2

Units may be placed by simply drawing a rectangle, clicking on the appropriate echelon, and unit role buttons, and specifying the unit designation (Figure 4).

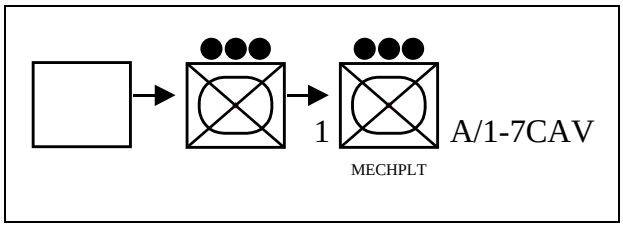

Figure 4. Unit Placement

Rather than placing additional data requirements on the user, the toolbars actually simplify the process by providing short cuts for drawing and designating units. Next the control measures, obstacles, and minefields need to be placed. As with units, a simple shape such as a line or rectangle is drawn first, the appropriate control measure button is clicked, the echelon (if any) is selected, and appropriate text is entered to designate the control measure (Figure 5).

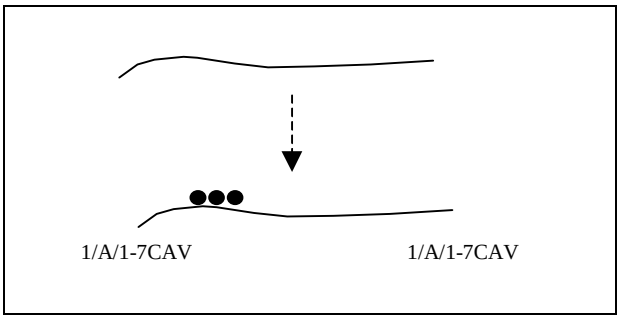

Figure 5. Tactical Boundary

As with unit placement, the toolbars actually simplify the process by providing shortcuts for drawing and designating the control measures.

Continuing the steps for placing control measures and units, a complete overlay may be developed in a very short period of time (Figure 6).

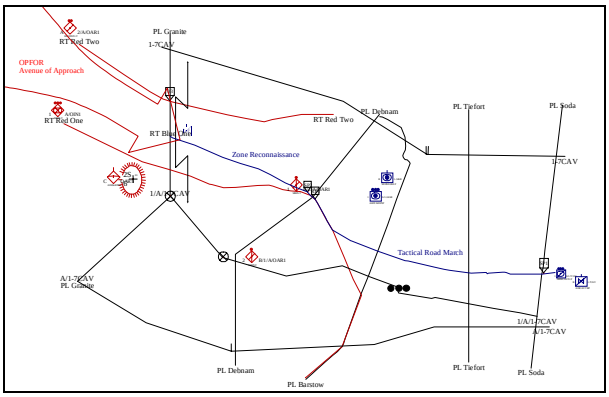

Figure 6. Example PowerPoint Overlay

Once the overlay is complete, the exercise data can be exported to a file in an ASCII format supported by CCTT and placed on a diskette. After the data have been saved to diskette, the exercise may be imported at the CCTT Master Control Console (MCC), and saved in CCTT's native exercise file format. This is the only simulation-specific step that must be taken by the training developer to transfer the exercise plan into the CCTT simulation environment. This example demonstrates a simple approach, using familiar applications that enhance and shortcut the time normally taken during the planning process to achieve the same end.

CONCLUSION

In review of the information presented, we can see that the development cycle for an integrated PC planning tool varies little from the development cycle of a specific simulation environment's native planning tool. A primary difference is the transition from the simulation environment's own programing language to the language of the user's desktop planning environment. This is a painful transition at best. Simulation developers have a preferred programming language. Software engineers learn to think in terms of their primary programming language. It is difficult to transition from one language and development environment to another. Educating an engineer, who is experienced in desktop development on a simulation development environment, is difficult as well. However, when new simulation environments depend on the development of planning applications, the desktop development environment already exists. Microsoft Office is an example of a desktop development environment, and these applications have been written. They have a broad user base and they work. Simulation developers can seize the opportunity to extend simulation into the familiar tools associated with the PC desktop with this user base. Recall the assumption "The basic training process does not vary by training environment." By integrating initialization data into the simulation domain from existing tools of the training domain, the views of both the training developer and simulation developer move that much closer to a uniform view of simulation and of training, meeting the needs of both.

REFERENCES

REFLECTONE Inc., AcuSoft Inc., (February 1999) Final Report – CCTT Interoperability Program – Initialization Enhancement.

Jeff Abbott, Dr. Mona Crissey, Mr. Gene Haga, (March 1999) Functionality of Exercise Initialization from the PC (Windows) Desktop, Simulation Interoperability Standards Organization, 99S-SIW-119.

Loral/Army Integrated Development Team, (January 1996) Software Design Document and Interface Design Document for the CCTT Unit Off-Site Planner.