Proceedings of the 1995 Winter Simulation Conference
ed. C. Alexopoulos, K. Kang, W. R. Lilegdon, and D. Goldsman

AFTER ACTION REVIEW SYSTEM (AARS) DESIGN AND FUNCTIONAL CAPABILITIES

Joseph W. Gibson

Logicon RDA
465 North Halstead, Suite 115
Pasadena, California 91107, U.S.A.

ABSTRACT

Logicon RDA has developed the Corps Battle Simulation After Action Review System (CBS AARS) to support the Battle Command Training Program (BCTP) requirement for training based on CBS-supported command post exercises (CPX). The CBS AARS is fielded and in use worldwide, and consists of the following subsystems: data collection, archive data storage, system and network management, analyst workstation, and presentation system. The analyst workstation is used to generate After Action Review (AAR) products. The presentation system includes advanced audio and video capabilities. It hosts the actual AAR presentation and provides advanced multimedia conferencing and display capabilities. The system also supports preparation of the Final Exercise Report (FER) and Professional Sustainment Packages (PSP). This paper describes the capabilities of the current system and the methodological issues related to its design and development. Emphasis is placed on identifying significant factors in the success of this development effort, and on clarifying problems overcome and lessons learned in the development process.

1 INTRODUCTION

In any command post exercise, training effectiveness depends on the collection, interpretation, and presentation of information that describes the course of events and related causal factors. The AAR is the presentation phase of this process. In simulation-supported CPXs, the collection process is automated, the interpretation is performed by analysts, and the AAR presentation graphics are generated by analysts using automated support tools.

Automating the AAR system for CBS exercises has increased the quality of the feedback provided to training audiences. The development of the CBS AARS presented significant challenges in that no comparable system existed to serve as a model. In addition there was no formal statement of requirements for the system. Hence an evolutionary development approach was adopted, incorporating inputs from three principal groups: BCTP analysts, BCTP training staff, and selected training audiences. Figure 1 depicts the high-level architecture for CBS AARS.

2 SYSTEM REQUIREMENTS

The functional requirements for the CBS AARS were developed from actual BCTP operational experience and successively evolved to support exercises of increased scope and complexity. The principal data collection requirements are as follows:

- All data available from CBS must be collectable.
- Actual data collected must be adaptable to support varying exercise training requirements.
- Data collection during exercises must be fully automated and require minimal operator intervention.
- Collected data must be continuously available online during the exercise for the analysts' use.

The system must produce a broad set of AAR graphics products based on the collected data. This allows the analyst to select the type of graphic that best presents a particular training point. Among the products supported by the fielded system are the following:

- Tactical situation displays based on Defense Mapping Agency (DMA) data sets, with CBS-generated (and other) information plotted as one or more overlays.
- Time-stepped animations of situation displays plotted on geographic maps with full play/pause/fast-forward single-step control.
- Common forms of presentation graphics based on office-automation software, including spreadsheets and business graphics such as bar and pie charts.
- A sophisticated graphics overlay drawing capability providing libraries of common military symbology and tools for developing custom symbol sets.

The analyst workstation must generate standard AAR data products relevant to training objectives for each Battlefield Operating System (BOS). The total number of products in use in CBS AARS is quite large and growing. Some samples are shown in Table 1.

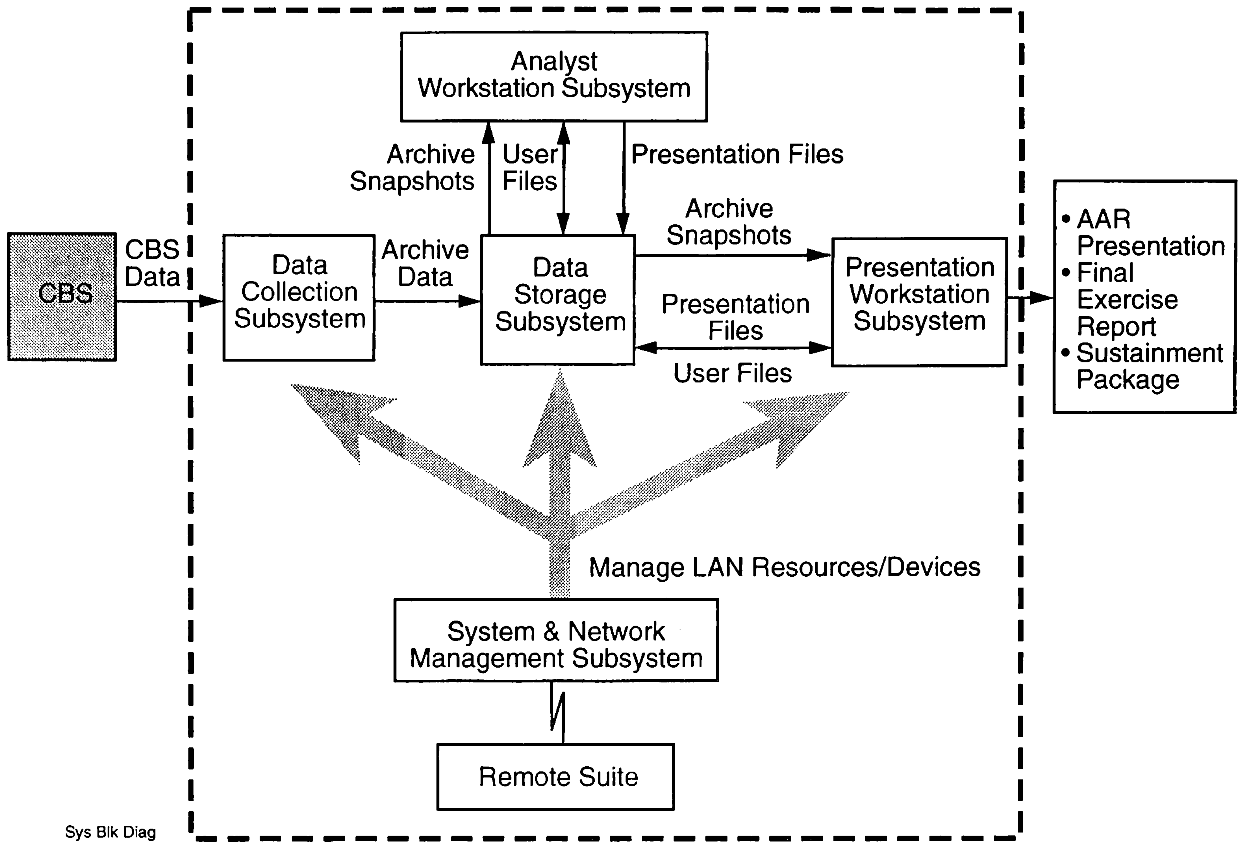

Figure 1: CBS AARS High-Level Architecture

Table 1: Sample BOS Data Products

BOS	Report
INTEL	See the Battlefield
(Shows difference between reality and perception)	
Maneuver	Correlation of Forces
(Shows weighting of main effort)	
ADA	Red Air Kills
(Bar chart: red sorties vs. kills)	
SOF	Deep Operations
(Shows effects of Special Forces deep operations)	
Fire Support	Volume of Fire
(Shows management of available assets)	
Mobility & Survivability	FASCAM Usage
(Histogram to show usage vs. capability)	
Command & Control	Orders Receipt
(Tracks order receipt in key HQs)	
Other Products	KIA vs WIA
(Shows ratio of KIAs to WIAs)	

Map graphics products must support plotting of simulation-derived data. The analysts' user interface must facilitate rapid and accurate development of presentation products. The interface must be simple, consistent, and graphical, and it must automate common and time-consuming functions, so that the analyst can focus on data evaluation rather than software graphics.

AAR hardware and software travel all over the world to support CPXs at designated sites. The products must be portable, robust, easy to set up and break down, and simple to configure for different training purposes.

Since CBS CPXs are normally conducted with the blue force (BLUFOR) analysts and training audience located at a forward training site and the battle simulation and opposing forces (OPFOR) analysts located at the National Simulation Center (NSC), it is essential that CBS AARS support distributed processing via high-bandwidth WAN connectivity between suites.

3 TECHNOLOGY SELECTION

The fielded system meets all of these requirements, and is regularly updated to conform to changes in CBS. Figure 2 depicts the high-level software design of CBS AARS. The

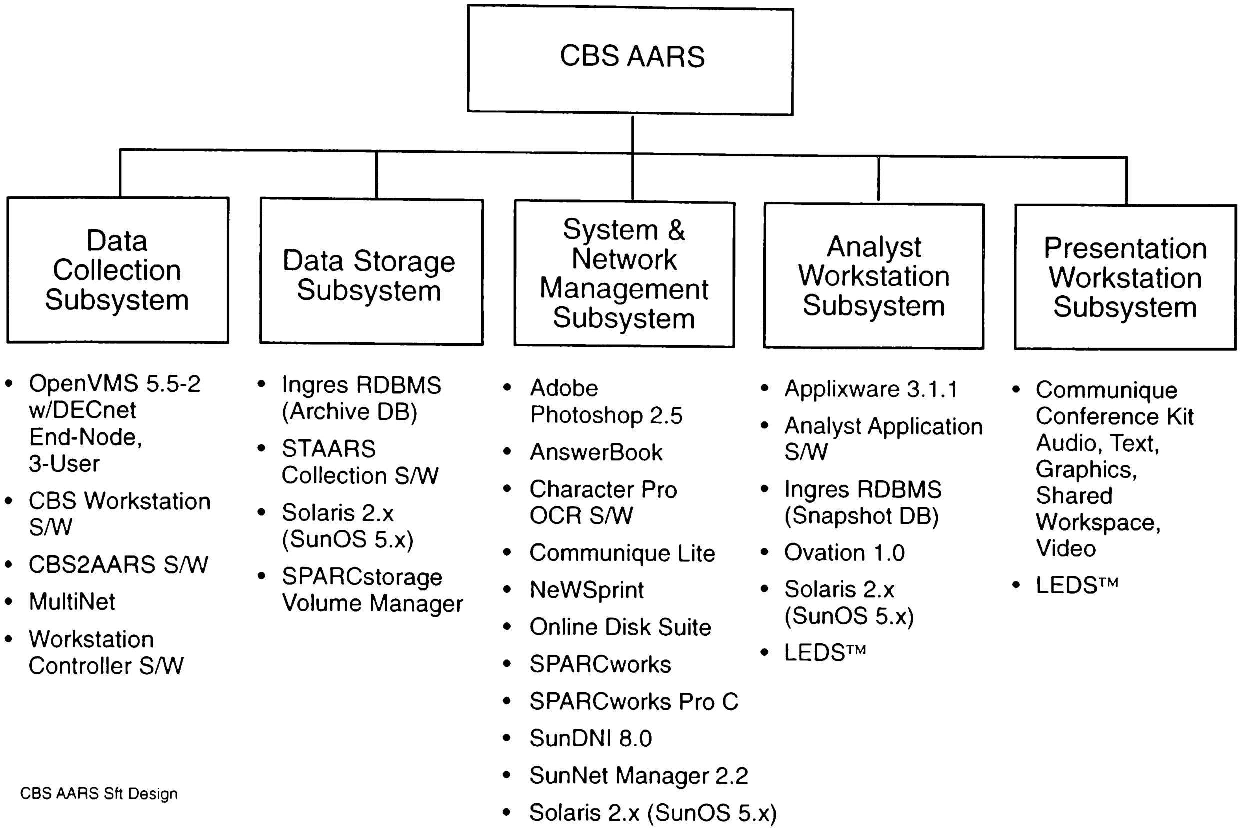

Figure 2: CBS AARS Software Design

design approach adopted was to host the AAR system on a state-of-the-art commercial UNIX network, while making maximal use of non-developmental items (NDI) and commercial off-the-shelf (COTS) software support. This strategy was adopted to minimize the amount of custom software required. A key part of this strategy was the adoption of the Logicon Environment for Decision Support (LEDS™), a COTS product developed by Logicon independently, as the central element in both the analyst and presentation workstation designs.

Development was carried out on the open, non-proprietary UNIX operating system, and as a result, the existing system is readily portable at modest development expense to most UNIX environments.

COTS products were employed wherever possible to speed up the development and fielding of useful capabilities, such as office automation tools and DBMS support. Integration of selected COTS products required that developmental software exploit the maximum capabilities of these products, but also, through a carefully structured interface, support rapid conversion to a different COTS offering if necessary.

We recognized early in the design process that, if successful, CBS AARS will remain in service for a long time, while computing technologies change rapidly. Thus, selection of supporting technologies was a critical issue and the design had to facilitate adaptive change. The CBS AARS development team has continued to survey commercial offerings for use in CBS AARS, and occasional changes have been made in COTS selections to acquire better functionality and reliability or lower cost.

The design that was chosen adhered to existing and emerging industry standards such as POSIX, XWindows, SQL, and TCP/IP. Development was based on published military standards such as DMA terrain data sets and FM-101-5-1 symbology. The C programming language was used because it allows for rapid code generation, and is non-proprietary, well-known, and portable.

4 SOFTWARE DESIGN

The developmental software for CBS AARS has a two-tiered structure. The first tier consists of a set of modular libraries, each implementing a specific, well-defined enabling technology. It includes libraries for the graphics toolkit, remote procedure call (RPC)-based interprocess communications, graphics overlay management, Universal Transverse Mercator (UTM) grid drawing, DMA data set management and map drawing, and graphical user interface (GUI) management.

The second tier contains software that implements the executable programs employed by analysts and support personnel. Each of these programs incorporates and employs several of the modular libraries from tier 1 to implement common functions.

These libraries have dependencies among themselves, being designed to be interoperable, mix-and-match functional components that developers could employ as needed. Thus any particular functionality need be implemented only once within the AAR software.

Where clear standards were available, compatibility was maintained with well-established public-domain products like the MIT X distribution, Standard C, and BSD UNIX. Where no clear standards were available, critical functions were reimplemented to insulate the AAR system from technology change and to guarantee reliability and access to the source code. For example, the Geographic Information System (GIS) has its own internally-defined data formats, and external data sources are preprocessed into the required formats for use within CBS AARS.

Finally, user access to COTS products was channeled through well-defined interfaces to support rapid conversion to new COTS products when necessary. An example is the use of different relational data base management systems (RDBMS) for simulation data storage.

5 SOFTWARE METHODOLOGY

Developing a complete and detailed set of requirements for the system was not feasible, so an evolutionary (spiral) development model was adopted. The goal for the initially fielded system was to make it useful enough to attract an engaged user community. Feedback obtained from the user community, operating this system under field conditions, was used to develop requirements for stepwise refinement of the system towards a final finished product. In this development model, one does not carry development to some final set of predetermined requirements, but ceases development when it is no longer cost-effective. Figure 3 depicts the evolution of the system.

Development was conducted on six-month cycles, each cycle based on the results of the previous iteration. Improvements delivered at each cycle were a mix of the following:

- Rapid prototype versions of initially fielded, new capabilities, generally at the request of the user community.
- Refinements and improvements of previously fielded capabilities, according to user requests.
- Bug fixes and developer-initiated engineering improvements.

Special care was taken to develop and maintain communication with the user community, as the feedback available in this way was considered critical to the success of the system. Whenever possible, prompt response was made to user complaints in order to maintain their interest in and commitment to the system. It is our judgment in retrospect that this dialog between the developers and the users was the single most important factor in the long-term success of the development effort.

The development team normally had 2-3 developers in Pasadena, CA, and 2 developers in Leavenworth, KS; each site had a supervisor who devoted 50% of his time to the project. Developers at both sites were normally responsible for analysis, design, coding, testing and documentation, and were expected to perform site support. The evolutionary (spiral) development phase occurred over three and one-half years, and is now settling into maintenance mode and lower staffing levels.

The resulting system has two computer software configuration items (CSCIs), one of about 150K source lines of code (SLOC) and one approximating 50K SLOC. Rough productivity estimates for this effort are well in excess of 30 SLOC/day, a very good figure by the usual standards. This includes tailoring LEDS™ to CBS AARS requirements and developing the custom AAR data collection, data archival, and report generation software for CBS AARS.

The present system is used enthusiastically and has the reliability and robustness of most commercial software in the UNIX environment. While it is clear that the spiral development model is inappropriate for embedded mission-critical software, it is highly cost-effective and appropriate for a project like CBS AARS.

6 TECHNICAL DESIGN ISSUES

Major technical design issues are discussed in the sections that follow. These technical comments are based on our field experience with this system.

6.1 Map Data

We have generally standardized, where possible, on DMA data sources, but we have used public domain data and data from CBS when nothing else was available. We also support scanned raster images that we can geographically register for display at specified locations. As previously mentioned, all data is preprocessed into an internal format. This insulates us from the wide variety of formats used for input. It is always simpler to create a translator for a variant data set than it is to write a complete drawing toolkit for each one. Internal data formats are based on XWindows Xlib-level drawing functionality. This provides both efficient drawing and a highly-portable and maintainable representation.

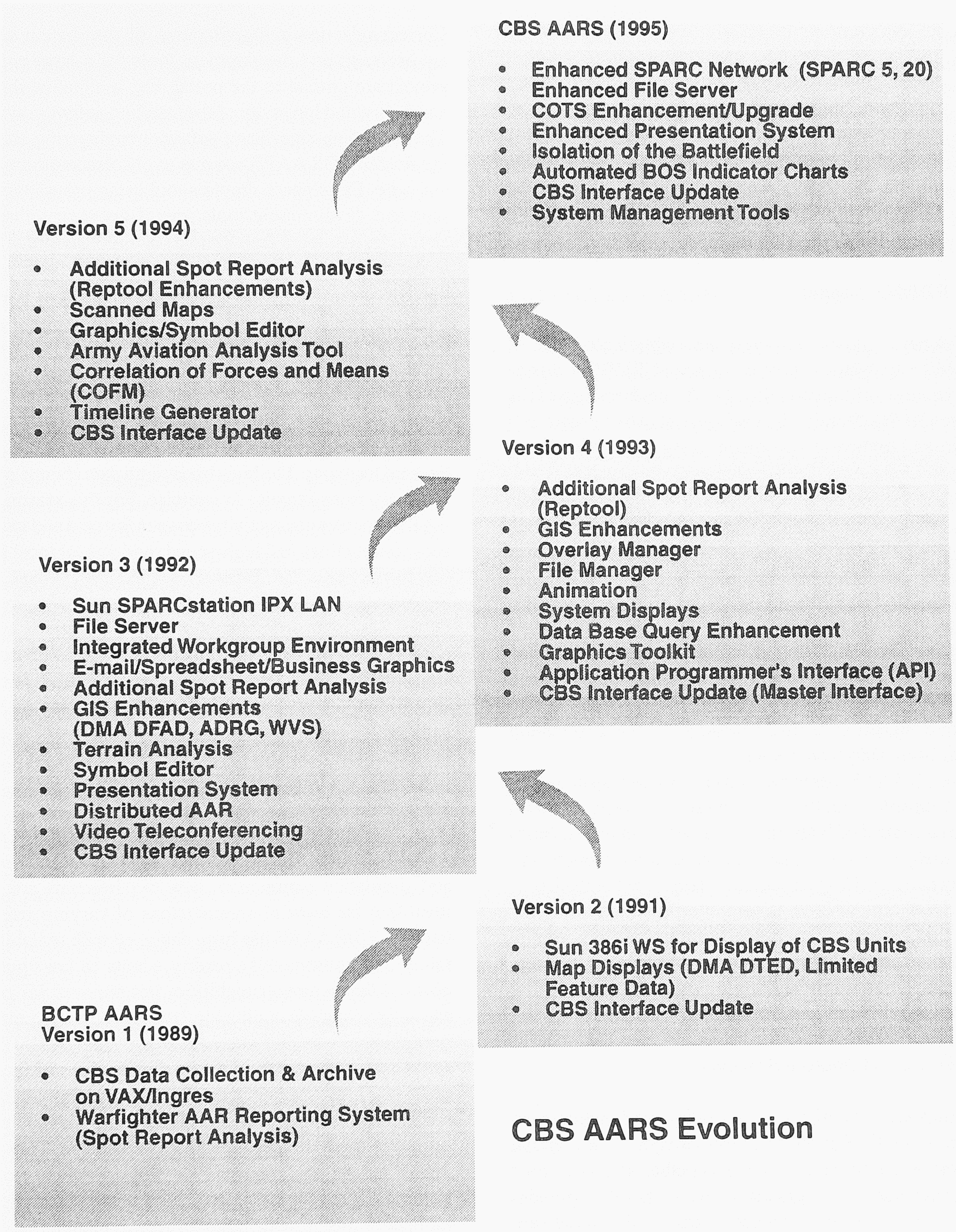

Figure 3: CBS AARS Evolution

6.2 Display Technology

Generally, the best possible color monitor is required. Color management is a continuing problem, and only eventual conversion to 24-bit color will provide a permanent solution. Resolution less than 1152x900 is inadequate. At resolutions of 1200x1024 and greater, interest in a bigger physical screen declines. However, introduction of virtual window managers offering larger-than-physical screen real estate with point-and-click selection has been very well accepted by the user community.

6.3 RDBMS Support

CBS AARS makes relatively unsophisticated use of RDBMS capabilities. Any commercial RDBMS would serve in terms of functionality, but the performance of the selected RDBMS under the load generated by the data collection and archive data storage subsystems is a critical issue for CBS AARS. Large volumes of data must be stored in synchronization with simulation time intervals, and this is the principal bottleneck in the present system.

6.4 The AAR Presentation System

The presentation system is a critical area, and the most dependent on available hardware support. Some of the greatest improvements in perceived satisfaction have been made in this area recently, based on the introduction of a custom-designed, state-of-the-art audio-visual presentation system from Maryland Sound.

The CBS AARS presentation audio-visual system supports multiple simultaneous inputs and outputs, with selective control of both. Input sources include live handheld video, VCR replay, and the presentation workstation display monitor. Outputs include a large front or back projection system and large-screen overflow monitors. This projection system offers much improved picture quality and allows larger viewing audiences. Video teleconferencing capabilities allow sharing of presentation material between the forward and rear sites.

6.5 Office Automation Software

We found wide variations in the quality and functional capabilities of COTS software in this category. Commercial products tested were not always acceptable, and we learned to test manufacturer upgrades to ensure that the upgrades are actually worth the conversion effort. We found reliability, consistency of the user interface, and presence of required functionality to be the critical issues.

6.6 Systems Software

Correct selection and application of the system support environment and development tools for a large software project are critical to the portability, maintainability, extensibility, and longevity of the resulting software systems. The remaining significant software engineering issues in the fielded CBS AARS are due to obsolescent supporting software. Developers should avoid using proprietary support and development tools, but if necessary, the tools should be insulated from the rest of the software with interfaces, so that support environment changes are relatively painless and inexpensive.

6.7 Networking Support

As noted in Section 2, CBS normally operates in a WAN environment, with portions of exercise personnel located in the rear, and training audiences located in one or more forward locations. The high-bandwidth data channels available vary from exercise to exercise. For this reason, CBS AARS provides connectivity between forward and rear sites using leased lines and high-speed modems, or local Ethernet connectivity. The Ethernet connectivity is used when the exercise forward site and the rear site are co-located. CBS AARS also supports front/rear connectivity via the CBS simulation network, but this is avoided because of the additional load. From a design standpoint, it is essential that AAR systems offer the widest set of WAN capabilities possible to support modern networked simulations.

6.8 Scalability

As CBS AARS gains increased visibility in the training and simulation community, new requirements are being identified for users at organizations of varying sizes and budgets. CBS AARS has been designed with scalability in mind, and consequently readily adapts to configuration changes that remove capabilities not required for certain users. For example, a division-level organization may not need the sophisticated presentation systems used by BCTP, as the audiences for their AARs are smaller and facilities are limited. In addition, the BCTP system has significant staffing requirements for operations and maintenance that cannot be supported by smaller organizations. Thus, CBS AARS can be configured as stand-alone, single workstation systems or for large distributed networks, and can be tailored to meet a wide range of requirements.

7 CONCLUSION

We would like to make the following points about what we found critical to the success of this software development effort.

For large, ill-defined software projects, a spiral development path based on intense interaction between the users and developers offers much improved chances of a well-accepted result. The trade-offs between quality assurance, traceability, and cost effectiveness must be scrutinized closely to ensure that an appropriate balance exists. In the context of spiral development strategies with heavy field feedback into the development process, a great deal of quality assurance and accountability derives from the suggestions and complaints of the field staff. A critical factor in this assessment is the cost of software failure in the fielded system. In the case of CBS AARS software, failure results in temporary unavailability of desired functions. The AAR can still proceed; hence the cost is small. In the case of CBS AARS, emphasis was placed on getting functionality in the field, rather than ensuring that all software was bug-free when fielded, with the assumption that the bugs would be found and corrected during use. This strategy has proven to be cost effective while still resulting in good software quality.

Using COTS in this system has proven cost effective and was a great time and work saver for the development team. We cannot emphasize too strongly the importance of careful selection of development tools. In spiral development strategies, work proceeds over several years. In the commercial computing world this may cover several generations of technology. Companies may rise and fall in that time. Without care about support technology, one can inadvertently produce an obsolete product, a product with no viable path for further development, or a product requiring painful and expensive conversion to maintain its viability.

The presently fielded system has been tested to military standards for compliance with government requirements. It thus demonstrates that rapid prototyping and iterative development do not preclude a robust and rigorously tested result. We expect it to serve well as the basis for future implementation of a standard AAR system that will offer support for exercises at all echelons in joint live/virtual/constructive-embedded training exercises.

ACKNOWLEDGMENTS

We wish to acknowledge the early and continuing support of BCTP for this development effort, and the support and guidance provided by STRICOM in more recent years.

We wish to especially acknowledge the great contribution made by the CBS AARS analyst teams and support staff. Without their continuing, detailed, and unfailingly direct criticisms, suggestions and complaints, the system would be much inferior to what it is today. We take great satisfaction in their enthusiastic use of CBS AARS.

AUTHOR BIOGRAPHY

JOSEPH W. GIBSON is manager of the Advanced Technology Department of the Systems Development Division of Logicon RDA. He holds an MSCS with emphasis on Information Systems Design from West Coast University, and a BA in Mathematics from Humboldt State University. He has contributed to the development and support of the Joint Theater-Level Simulation (JTLS), CBS and CBS AARS, and has over 15 years of experience in the design, development, fielding, and maintenance of computing systems. His technical interests include highly portable software development, MIMD parallel computing architectures, and the design of distributed computing architectures to support large-scale training simulations.