Converting a Large Simulation System to a 64-bit Computer

Roger D. Smith
Titan Corporation
Orlando, Florida
rdsmith@titan.com

ABSTRACT

Command and staff level training events and experiments rely on very large simulation scenarios. Events like Ulchi Focus Lens and Millennium/Olympic Challenge may be driven by databases with 10,000 objects, running 24-hours a day, for more than 7 days. These scenario databases have grown extremely large as our interests in representing a more diverse set of aggregate units and individual vehicles have increased. Support units like logistics and maintenance forces have been added; non-combatant units play a role; and UAVs and precision weapons are represented at the entity level. These factors have resulted in the repeated doubling of the scenario data size and have driven the current generation of 32-bit computers to the edge of their ability to support these events.

Legacy systems like the Corps Battle Simulation (CBS) rely on the most stable version of the Red Hat Linux operating systems running on an Intel-based computer. Both Red Hat 7.3 and the Pentium chip support 32-bit computing, which imposes limits on the size of memory, address space, and individual files that can be used by the computer or addressed by a single application. Simulations like CBS have evolved such large scenarios that they are beginning to exceed these limits.

In this paper we describe our experiments in porting CBS to a 64-bit computing environment. This transition included the impacts and limitations of a new CPU, operating system, device drivers, shared libraries, compiled executables, and language compilers. This process exposed a number of valuable and surprising lessons that will be faced by any project converting to a 64-bit computer. There are a number of other simulation systems that will soon exceed the limits of 32-bit computers and will face the need for this same type of conversion.

ABOUT THE AUTHORS

ROGER SMITH is a Vice President and Group CTO at Titan Corporation working on next-generation simulation applications and distributed computing technologies. His most current work has been creating techniques to model counter-terrorism, infrastructure protection, and the design of standardized modeling techniques. He is the creator and instructor for a series of simulation courses that have educated hundreds of simulation professionals. He serves on advisory boards for Sandia National Laboratories, University of Florida, and University of Central Florida, and on the editorial boards of the *ACM Transactions on Modeling and Computer Simulation* and the *IRI Research Technology Management*. He holds degrees in mathematics, statistics, computer science, business administration, and management.

Converting a Large Simulation System to a 64-bit Computer

Roger D. Smith
Titan Corporation
Orlando, Florida
rdsmith@titan.com

LARGE SIMULATION COMPUTING LIMITATIONS

Military simulation systems are used to represent all types and levels of operations in support of training, mission rehearsal, requirements analysis, and new concept evaluation. Like all computer applications, these simulations began as small applications that were limited by the maturity of the software industry and the computational capabilities of affordable computer hardware. However, over the last several decades, our ability to create very large simulations and powerful computers has allowed us to grow simulation systems to support military operations that are an order of magnitude larger than earlier systems.

Some of the largest current simulation systems are beginning to reach the limits of 32-bit computers and operating systems. Specifically, the Corps Battle Simulation (CBS) developed by the Program Executive Office for Simulation, Training, and Instrumentation (PEO STRI), has experienced growth in its computing algorithms and scenario sizes that can dangerously approach 32-bit limits. Current customer demands for more detailed unit and terrain representations threaten to exceed the maximum memory and file sizes dictated by 32-bit data representations.

"CBS is a geographically and functionally distributed air/land warfare simulation that drives the U.S. Army Battle Command Training Program's (BCTP) War Fighter Exercises as well as Corps and Division command training exercises for the active Army and the National Guard. The simulation also serves as the Land Warfare component of various Joint Training Exercises as a member of the Joint Training Confederation (JTC). CBS provides training stimuli for all ground forces staff elements from Brigade to Corps including combat, combat support, combat service support, and fixed and rotary wing air operations. All Battle Operating Systems are represented: Maneuver, Command & Control, Fire Support, Air Defense, Combat Service Support, Mobility / Countermobility / Survivability, Intelligence, as well as fixed and rotary wing air operations, NBC operations including Smoke and Chemical Recon and Decontamination, Special Operations, Civil Affairs and Psychological Operations." (Zedo, 2004)

CBS is currently fielded to a number of sites that purchase and configure their own computer systems. Therefore, an exact specification of the hardware and operating system for the simulation is not possible. However, Table 1 illustrates the specifications recommended for the system and a high performance configuration that has been used to test its performance with large databases.

Memory Limitations

The problems that are occurring with these configurations are primarily due to exceeding maximum addressable memory and file sizes. Theoretically, 32-bit computers provide a maximum of 2^{32} bytes of RAM and a corresponding maximum of a 2^{32} byte file size, which is approximately 4 gigabytes (GB) each. However, in practice, 4GB is the maximum available to an entire executable and must be shared between the developed software and shared libraries that are linked into the executables. To manage this

Table 1. CBS Computer Configuration and Scenario Sizes.

Configuration Item	Recommended Spec	High Performance Testing
CPU	Dual Pentium 1.1 GHz	Dual Pentium 2.8 GHz
RAM	2 GB	4 GB
Hard Drive	Dual 20 GB	Dual 40 GB
Operating System	Red Hat Linux v7.3	Red Hat Linux v7.3 with Kernel Mods
Shared Libraries	SIMSCRIPT II.5 v3.0	SIMSCRIPT II.5 v3.0
Scenario Size	10,000 Aggregate Units	12,000 Aggregate Units
Terrain Playbox	100 km^2	200 km^2

sharing, the Red Hat Linux operating system limits developed software to addressing 2.5GB of memory and allocates the remaining 1.5GB to the shared libraries. This prevents an application developer from incorrectly assuming that all 4GB's are available to his or her software, because some unknown amount must be left available to shared libraries that perform operations that could include graphics manipulation, networking, mathematics, and a host of other functions.

One of the shared libraries that CBS relies upon is the CACI SIMSCRIPT II.5 simulation programming language. Software that uses this library must also limit the size of all data and pointer values to less than 1 GB because that software uses smaller addresses to manipulate data. As a result, any data values within a CBS executable that will be passed into a SIMCRIPT routine must reside within the first 1GB of the computer's memory.

File Size Limitations

Red Hat Linux v7.3 imposes a 2GB limit on the size of a single file that can be created by an application. Within CBS, the largest file that is created is the Checkpoint/Modify/Restart (CMR) file, which captures all of the simulation state data at a given time and provides a stable point from which to restart the simulation in the event of a system crash or unacceptable deviation in the scenario. For scenarios with 10,000 units, the Checkpoint file has been measured at 1.5 GB. This number is dangerously close to the 2GB limit and poses a threat to the development of scenarios for large exercises. Since this Checkpoint file size is not known at the time of scenario development, there is a risk that the system could fail to perform an exercise in spite of the fact that the scenario is correctly structured and the software is operating correctly.

Note that the 2GB limitation on file sizes has been extended on Unix systems through the adoption of the "Large File Summit extensions" (Josey, 2004). This may allow addressing up to 1 terabyte – depending upon the implementation (Mauro, 1998).

Computational Performance

CBS is using some of the fastest Pentium class CPUs available. The migration from the older dual 1.1 GHz speed to the new dual 2.8 GHz speed has been essential to support the large scenarios that are being called for by the customer. The 1.1 GHz chips are not capable of running a 10,000 unit scenario in real-time, as required for training exercises.

Faster chips like the 2.8 GHz machines that are being used for the largest scenarios provide a temporary solution for the computational problem. However, to maintain a safe buffer of performance for peak periods and to support high fidelity unit and terrain representations, CBS will have to continue to migrate to faster computers or explore the potential of moving to a scalable multi-processor (SMP) machine with many more than the current dual processors – perhaps as many as 32 processors. There have been discussions of porting the simulation to an SMP machine. However, the purchase and support costs of these are higher than for a more standard dual processor workstation and the expertise necessary to operate them may not be available at all of the sites at which CBS is fielded. Since CBS is a mature system, there is not sufficient financial support to make such a drastic improvement in the hardware environment for the system.

DEFINING 64-BIT COMPUTING

The affordable solution for providing sufficient computational power to support CBS and the evolving large scenarios appears to be rehosting them on the next generation of computer chips, systems, and operating systems. This generation is usually referred to as 64-bit computing because the computers are able to process data in much larger sizes – specifically 2^{64} byte sizes, or approximately 18 exabytes. This is accomplished by creating computer chips that contain both 64-bit registers and a 64-bit data path. This allows the CPU to handle a number that is as large as 2^{64} and the data paths to deliver that size of number to the CPU for computation.

Once such a CPU is acquired, all of the software that resides on the computer, beginning with the operating system and device drivers, must be re-built with new data sizes on all of the variables. This enables applications to access and manipulate data items that are up to 64-bits wide.

Current 32-bit application data follows a data model called ILP32, which means Integer, Long, and Pointer data types are 32-bits long. 64-bit computers have gone through a two-step evolution in which they first used LP64 and then ILP64 data types (Open Group, 2004). The first standard applied 64-bits only to Long and Pointer types and the second added Integers to this (Hewlett Packard, 2004b) (Microsoft, 2004). The size of data in each of these standards is shown in Table 2.

Table 2. Sizes of Data Types in ILP32, LP64, and ILP64 Data Models.

Data Type	ILP32	LP64	ILP64
char	8	8	8
short	16	16	16
int	32	32	64
long	32	64	64
long long	64	64	--
pointer	32	64	64
float	32	32	64
double	64	64	64
long double	128	128	128
enum	32	32	64

These two changes in the hardware handling of data enable the operating system and applications to expand their maximum capacity on addressable memory and file sizes. They also potentially improve the computational performance for algorithms that are performing mathematics with large integers. From the perspective of a user, the advantages of 64-bit computing are expressed as three capabilities:

- Larger Addressable Memory – applications will be able to address memory beyond the 4GB limit of the 32-bit CPU and the 2GB limit of many operating systems,
- Large File Size – applications will be able to read and write files that are larger than 2GB,
- Large Integer Arithmetic – applications will be able to perform computations using larger and/or more precise integers,
- Computational Speed – some CPU/OS combinations may be able to perform multiple 32-bit operations simultaneously. (Hewlett Packard, 2004a & 2004b) (Mainelli, 2003) (Thanawala, 2000)

These capabilities are the advantages that CBS is seeking to allow it to continue to meet growing customer demands. However, many authors have observed that 64-bit computing is not necessary for most applications and that a complete conversion to these computers will be very long in coming (Coursey, 2003) (Intel, 2004) (Mainelli, 2003). This is significant because it means that most complex systems will be integrating both 64-bit and 32-bit computers together for many more years.

Quantifying 18 exabytes

A 64-bit computer will enable the addressing of up to 18 exabytes of memory. However, no computer in the foreseeable future will be able to insert this much memory into a machine. In fact, it is unlikely that 18 exabytes of memory currently exists on the planet. This number is so large that analogies are the only means of putting it into perspective. Several of these analogies will compare this extremely large number to the 4GB limit on addressable memory that exists in a 32-bit computer.

> *Analogy 1:* If we were to represent 4GB as a line that is one inch long. Then the corresponding 18 exabyte line would be 71,023 miles long.
> *Analogy 2:* A British 2 pence coin covers an area of approximately 1.7 cm^2. If this coin represented 4GB of memory, then 18 exabytes would be an area the size of Scotland (approximately 78,133 km^2).
> *Analogy 3:* The current world population is approximately 6.3 billion people. 18 exabytes is enough memory to give every living person a computer with nearly 3GB of RAM inside.
> *Analogy 4:* If a technician were to attempt to insert 18 exabytes into a computer using 1GB DIMMs, inserting one each second, working 8 hours/day, and taking no breaks; then that technician would complete the job in 3,651½ years. To deliver one computer in 2004, that technician would have had to begin the task in the year 1648 B.C.

It is clear that 18 exabytes is an unbelievably large amount of memory and that no computer in the near future will even approach this limit. Vendors like Red Hat have settled on a more practical limit for the 64-bit version of their operating systems. Red Hat Enterprise Linux 2.1 (their 64-bit product) will support 64GB of memory.

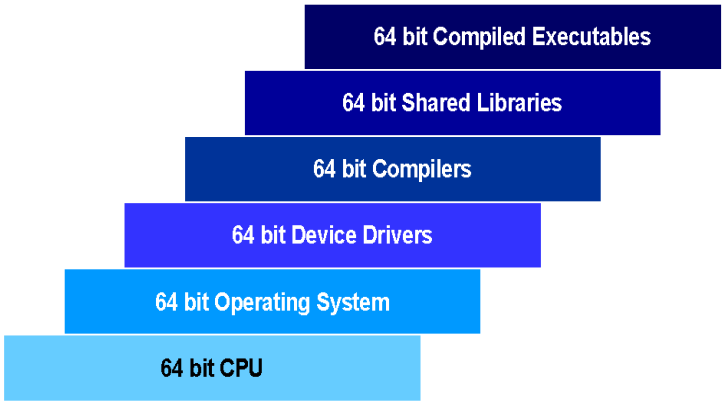

Figure 1. Layered View of System Dependencies

Layered View

The deployment of 64-bit computers to a wide audience of users is going to depend upon the availability of a number of hardware and software products. Many simulation systems, including CBS, rely upon commercial libraries of software that must be ported to 64-bits before the application itself can make the transition. Figure 1 illustrates the stacked and dependent nature of the problem.

The new CPU is the first step in this climb from 32- to 64-bit computing. This must be followed by a 64-bit version of the operating system and 64-bit device drivers for each operating system. System vendors have been eager to create the necessary device drivers for their hardware. For example, we found that Hewlett Packard had already developed all of the device drivers necessary for the graphics cards, sound cards, hard drives, CD and DVD-ROM drives, and other devices that they configure into their machines. This package is called the "HP Enablement Kit for Linux on Itanium 2 Processors".

Following the device drivers, 64-bit compilers must become available to create the shared libraries and executables. Red Hat Linux WS 2.1 includes the GNU compiler in 64-bit mode. However, specific library and application vendors may require new compilers from specific vendors, delaying the deployment of 64-bit versions of their software.

32- to 64-bit Data Transfer

All of the computer vendors that we studied expected the transition to 64-bit computing to be a long process. Therefore, they are creating systems that are able to execute both 32- and 64-bit applications in the same environment. The chips and operating system will recognize both data sizes and instruction sets. This essentially creates a 32-bit sandbox that can accommodate all current executables while the conversion to 64-bit is occurring. Therefore, it is possible to continue using 32-bit applications on a new 64-bit computer without recompiling them – as long as you also retain all of the 32-bit shared libraries that the application uses.

However, in general it is not possible for 64-bit and a 32-bit software to interoperate with each other. This is driven by two fundamental differences in the representation of data. First, a 64-bit application will represent a `pointer` or a `long` data type as a 64-bit value. If that value is passed to a 32-bit application, operating system, or shared library the value will be truncated to 32-bits resulting in an error. This issue is one reason that the LP64 and ILP64 data models shown above exist. The LP64 data model attempts to maximize the interoperability between 32- and 64-bit software. The ILP64 data model attempts to create a more complete 64-bit data model for future computing. Each vendor is determining whether to use LP64 or ILP64.

The second issue for interoperability stems from the first. Most applications make use of data structures that combine variables of different data types. These structures contain offsets to align data boundaries and these offsets are different for 32- and 64-bit applications. Figure 2 provides an illustration of this in which the structure `foo` contains four different data types. Compiled for 32-bit computing, this data structure is 24 bytes long. Compiled for 64-bit computing, the same data structure is 32 bytes long and has different data item boundaries (Hewlett Packard, 2004a).

These differences in the data models result in erroneous data when 64-bit and 32-bit applications

```
typedef struct foo {
      char  key;
      long  val;
      int   id;
      char  sig[10];
} FOO;
```

Name	Data Type	ILP32 Offset	LP64 Offset	ILP32 Size	LP64 Size
key	char	0	0	1	1
val	long	4	8	4	8
id	int	8	16	4	4
sig	char[10]	12	20	10	10
			Total Size	24 bytes	32 bytes

Figure 2. Example of 32- and 64-bit Data Structure Differences.

exchange data. There are several solutions that can make this data exchange work correctly. The most practical is to use eXternal Data Representation (XDR) library to filter all data exchanges and create representations on both machines that are appropriate. A programmer can also choose to pad data manually, carefully select matching LP32 and ILP64 data types, or convert all data to be exchanged into ASCII characters.

Advantages to the Simulation System

As described earlier, there are four major advantages of 64-bit computing. Each of these can benefit the deployment of the Corps Battle Simulation. The size of CBS scenarios and the resolution of CBS terrain databases have grown so significantly in recent years, that the system is currently on the very edge of the maximum memory and file sizes of a 32-bit computer. The system also contains several computations that are dangerously close to overflowing 32-bit data types.

Extended Memory

Though 32-bit computers can address 4GB of memory, the Red Hat operating system divides that into space allocated to the application code and space for shared libraries. The current configurations allow an application to use only 2.5GB of memory and reserve the remaining 1.5GB for the shared libraries. CBS scenarios reside in memory during execution and those memory images are already at the 1.5GB level for many exercises. The scenarios for the Korean Ulchi Focus Lens (UFL) exercises are the largest and can exceed 2GB. Both the Jet Propulsion Laboratory (JPL) and Titan Corp. have expended considerable effort to reduce the memory footprint of a scenario. These have resulted in improvements on the order of 10MB, which is not enough to provide a sufficient safety buffer for the system.

Scientists at JPL have also modified the Red Hat Linux kernel to move the 2.5/1.5GB memory boundary and provide more space for the application. These modifications work well on a defined hardware and operating system configuration. Unfortunately, the CBS system is fielded on a number of different computer configurations around the world and the kernel modification is not compatible with all of those. Therefore, this modification has been removed from the deployed system.

Customers are also pressing for CBS to increase the level of detail in the terrain database. Since the CBS terrain playbox may be as big as 200 km^2, any increase in fidelity results in major increases in the memory required to operate the system and the disk drive required to store the data prior to loading.

CBS relies on the commercial SIMSCRIPT simulation language produced by CACI Inc. This language provides an extensive number of shared libraries for queuing services, mathematics, statistics, and random variates. The SIMSCRIPT shared libraries are configured such that they cannot access memory over 1GB. This creates an additional limit to some parts of the CBS software. It is currently not possible for the government contractors to modify the SIMSCRIPT product to overcome this limit. But, the government is working with the vendor to modify this limit and release a new version of the product that can be deployed with CBS. Those negotiations include more extensive support for a 64-bit computing environment.

Large File Sizes

The initial scenario for a CBS exercise is divided into multiple files that generally do not approach the 2GB limit imposed by the standard Red Hat services. However, during the execution of the simulation,

complete memory images are dumped to a single file as a "checkpoint". The checkpoint file represents a reliable state of the simulation at a given time. During long and involved exercises, these historical checkpoints are very useful for recovery of a system that has crashed and all data in memory lost. Checkpoints also provide points of departure for exploring vignettes that were not pursued in the major exercise, or points from which an exercise can be redirected if the original execution led to a poor conclusion. Because the CBS checkpoint file can be used for so many purposes, it is referred to as the Checkpoint, Restore, Modify (CRM) file.

Currently, CRM files are being created that are 1.5GB in size. This is within 500MB of the maximum limit of the system. Therefore, there is some concern that as scenario sizes and terrain details increase, the CRM files will reach the maximum file limits. This issue could also be resolved through major modifications to the CBS software. But, the development, debugging, and deployment of such global changes are estimated to have a significant impact on the schedule of the program.

Large Integer Mathematics

64-bit computers allow applications to work with much larger numbers than can be handled by a 32-bit computer – specifically large integers. This improvement is useful for a relatively small customer base that is involved in detailed scientific computing. Applications like environmental modeling, nuclear weapons testing, computational fluid dynamics, and corps-level command and staff simulations are a few of these.

CBS has encountered problems in the past when combining very large and very small numbers within a single algorithm. These problems include a computational mathematics classic in which a very large number is divided by a very small number, which can lead to quite unpredictable erroneous answers. For example, CBS represents ammunition as a ratio of the total vehicle weight – specifically 0.00002. Therefore, when the system attempts to calculate the number of rounds of ammunition in an entire corps, it must divide the summed total weight of all vehicles in the corps by the ammunition-to-vehicle weight ratio. This is a very large number of tons divided by the very small ratio 0.00002. On the old VAX computers these calculations were known to result in errors for very large units. Reliability in this and other similar calculations has improved in the Intel/Linux environment, but both JPL and Titan remain uncertain that it is being performed correctly in all cases. Given situations like these, CBS can rely upon the improved large number mathematics to eliminate the uncertainty associated with these types of calculations.

Parallel 32-bit Computation

Most vendors recognize that the large majority of their customers do not require the large integer mathematics described above. Even those that do will often take advantage of it for only a few calculations, which is the situation with CBS. Therefore, rather than allowing this computational power to lie fallow, they have designed the systems to use the 64-bit register sizes to perform two 32-bit computations in parallel. For some applications this has the potential to nearly double the speed of processing during computationally intensive operations with small numbers (Coursey, 2003).

CBS can make much more ready use of this parallel computation. The simulation performs an extensive number of calculations of line-of-sight, movement rates and distances over terrain, and equations of combat attrition. In a scenario of 10,000 units these computations represent a significant amount of the workload on the computer and parallel 32-bit calculations of these may significantly improve the performance of the system.

SYSTEM CONFIGURATIONS

The necessity of achieving compatibility across all of the layers shown in Figure 1 led us to the decision to purchase computer systems from a vendor that would supply as many compatible 64-bit layers as possible. Most of the components for constructing a 64-bit computer are available from parts vendors on the Internet. The 64-bit version of the Red Hat operating system is also available for purchase or download. Red Hat also includes a number of device drivers and we may have been able to purchase hardware for which 64-bit drivers are readily available.

In spite of this, we determined that the purpose of this project was to focus our time, effort, and money on the rehost of the military simulation system and not on building and supporting a new generation of computer equipment. As other projects report their success with different hardware, operating system, and software components we will examine those systems in order to identify one with better cost/performance ratios.

Because we believe that the best process for porting a 32-bit application is to move the 32-bit executables to a 64-bit computer running in a 32-bit mode, the preferred machine for the conversion is one based on the AMD Opteron chip. The AMD chip can run both

Table 3. Variety of 64-bit Computer Systems.

Vendor	H-P	UNISYS	Sun	IBM	IBM	SGI	Apple
System	zx6000	Aries 410	Fire v250	RS/6000 ES S70	IntelliStation A Pro	Altix 3000	Mac G5
CPU	Itanium 2 Dual 1.5GHz	Itanium 2 Quad 1.5GHz	UltraSPARC III Dual 1.28GHz	PowerPC	AMD Opteron Dual 2.2GHz	Itanium 2	PowerPC Dual 2.0GHz
Operating System	Red Hat Enterprise WS 2.1	MS Windows Server 2003, Enterprise Edition	Solaris 9	AIX v4.3	Red Hat Enterprise WS 3	Red Hat Enterprise WS 2.1	OS/X Panther
Max RAM	24 GB	128 GB	8 GB	16 GB	16 GB	4 TB	8 GB

32- and 64-bit applications simultaneously. This capability is not currently available for the Intel Itanium 2 chip. Beyond Intel and AMD, there are a number of other chips and system configurations available. Table 3 illustrates some of the variety of 64-bit computing environments that are available from full-system vendors.

EVOLVING TO A 64-BIT ENVIRONMENT

We have determined that the best approach to porting a 32-bit application to a 64-bit environment is to do so in two phases. The first phase is a rehost of all existing applications and libraries to the new computer. The second phase is the recompilation of applications and libraries, and the creation of new device drivers as necessary.

32-bit Rehost

The core simulation engine for CBS is the Game Engine Executive Process (GEEP). This is the application that manages most of the military units and includes most of the combat event algorithms. The GEEP is also heavily dependent upon the SIMCRIPT programming language and libraries described earlier. Rehosting this application to a 64-bit computer requires moving the GEEP and the SIMSCRIPT libraries to the new computer – without recompiling them. The new machine will also host the client software for the relational database that stores the scenario data.

During the rehosting phase we plan to demonstrate that the new computer system can run 32-bit applications unchanged. One key to achieving this is the availability of device drivers to support the hardware included in the system. That is one important reason for our preference for purchasing a complete system from a vendor rather than building the system from parts.

In addition to testing the stability of the system in the new environment, we plan to conduct performance tests to determine the speed of the applications in the new environment versus their speed on 32-bit computers. The high performance configuration for 32-bit CBS computers currently uses 2.8GHz Pentium chips. Therefore, we do not expect to achieve faster execution performance with this first configuration. There are multiple mitigating factors:

- Slower chip speed,
- No recompile to take advantage of 64-bit capabilities,
- Operating in a 32-bit emulation mode, and
- Early versions of operating system not optimized.

Though we do not expect a performance improvement with this configuration, we do want to capture performance data that can be compared with the configuration created in phase two of the project.

During this phase we can also conduct experiments with the Red Hat GNU compilers to identify unforeseen dependencies within CBS on commercial libraries, device drivers, or other applications. These compilation experiments are not intended to create a working 64-bit executable, but are part of the preparation for the next phase.

64-bit Recompile

The recompile of the CBS executables is dependent upon the availability of commercial software libraries that have been built for a 64-bit version of Red Hat Linux. Specifically, we are waiting for CACI to create a 64-bit version of SIMCRIPT. Once that is available, we can begin to build and test 64-bit simulation executables. As described above, our compilation experiments will probably expose additional dependencies that must be dealt with prior to a successful 64-bit compilation. At this point we do not know what those dependencies are and cannot describe them here.

The SIMSCRIPT language is designed to make it easier to create simulation programs that rely on different types of queuing, statistical distributions, and random variates. A sample of a SIMSCRIPT program for a basic queuing system is shown in Figure 3 to emphasize its unique structure from more common C++ or Java programs (Hughes, 2003). Most of the commands in this language invoke queuing or statistical routines that are part of the commercial product.

The degree of performance improvements under the recompile is not clear. The CPUs in the selected computers are 1.5GHz machines, which are faster than many of the hardware configurations used to run CBS at field sites. But, these are slower than the 2.8GHz machines that have been used to test performance at JPL. Therefore, we are not certain how much improvement will be gained through the Itanium architecture, faster internal buses, and potential parallel computations.

Field Deployment

CBS is a system fielded to over a dozen operational sites. It is the core training software at the Army's Battle Simulation Centers in the U.S., Europe, and South Korea. Therefore, the success of porting the system to a 64-bit computer is measured by the degree to which it improves training in the field. It is clear that the phase one rehost version of the software will not be fielded. Phase one is a risk management step to improve the probability of success in phase two. Following phase two recompile, the system will undergo extensive testing at two different sites before being deployed to a battle simulation center. The field deployment of a 64-bit version of CBS will require at least two years of experimentation, development, integration, testing, and fielding. Such a significant change also represents a major investment on the part of each of the battle simulation centers that purchase and maintain their own hardware systems.

64-bit CBS is a very forward-looking effort that begins addressing a potential problem before it severely impacts a major training event. This project will provide valuable knowledge to a number of other training systems that are experiencing performance limitations of a 32-bit computing environment. Most notable of these is the Army's Warfighter Simulation (WARSIM) that is the follow-on replacement for CBS.

CONCLUSION

Our investigations into the feasibility of converting CBS to a 64-bit computing environment were quite favorable. In fact, we are convinced that the move to 64-bit computing is inevitable for all large applications – scientific, database servers, and simulation. Like the evolution that occurred from DEC VAX, to SUN/SGI workstation, to 32-bit Linux PC, we believe that 64-bit computing is the obvious and beneficial next step.

At the time of this writing 64-bit computers are in use, but very sparsely. We were interested in studying

```
1       MAIN
2               READ MEAN.INTERARRIVAL.TIME,
3                       MEAN.SERVICE.TIME, AND TOT.DELAYS
4               CREATE EVERY SERVER(1)
5               LET U.SERVER(1) = 1
6               ACTIVATE AN ARRIVAL.GENERATOR NOW
7               START SIMULATION
8       END

1       PROCESS ARRIVAL.GENERATOR
2               WHILE TIME.V >= 0.0
3                DO
4              WAIT EXPONENTIAL.F(MEAN.INTERARRIVAL.TIME,
5                                       1) MINUTES
6                  ACTIVATE A CUSTOMER NOW
7               LOOP
8       END
```

Figure 3. Sample of a SIMSCRIPT Queuing Program (Hughes, 2003)

several cases of successful migration, but found few available. We anticipate that the biggest hurdle in migrating CBS and other applications will be the pace at which third party applications like SIMSCRIPT are compiled and marketed for the 64-bit operating system. Other hurdles will certainly become apparent as we get further into this project.

ACKNOWLEDGEMENTS

We would like to acknowledge the support of Dave Zedo at the US Army Program Executive Office for Simulation, Training, and Instrumentation in Orlando, Florida. We also received valuable comments from Tim Burke at Titan Corporation in Leavenworth, Kansas and Jay Braun at the Jet Propulsion Laboratories in Pasadena, California.

REFERENCES

CACI. (2003). "SIMSCRIPT II.5". CACI Web Site. Retrieved April 2, 2004, from http://www.caciasl.com/products/simscript.cfm

Coursey, Dave. (September, 2003). "Why you do (or don't) need a 64-bit computer". ZD-net web site. Retrieved April 2, 2004, from http://reviews-zdnet.com.com/AnchorDesk/4520-7297_16-5082999.html

Hewlett Packard. (2004a). "Interoperability of 32- and 64-bit Applications on 64-bit HP-UX". Retrieved April 2, 2004, from http://devrsrc1.external.hp.com/cgi-bin/STK/pfnew.cgi?in=/STK/partner/3264interop.html

Hewlett Packard. (2004b). "What is 64-bit computing?" Retrieved April 2, 2004, from http://h21007.www2.hp.com/dspp/tech/tech_TechDocumentDetailPage_IDX/1,1701,989,00.html

Hughes, Herman. (Fall 2003). "Modeling and Simulation with SIMSCRIPT II.5". Retrieved April 2, 2004, from http://www.cse.msu.edu/~cse808/

IBM. (2004). "The RS/6000 64-bit Solution". Retrieved April 2, 2004, from http://www-1.ibm.com/servers/eserver/pseries/hardware/whitepapers/64bit6.html

Intel. (2004). "Itanium 2 Processor". Retrieved April 2, 2004, from http://www.intel.com/products/server/processors/server/itanium2/index.htm?iid=ipp_browse+featureprocess_itanium2&

Josey, Andrew. (2004). "Data Size Neutrality and 64-bit Support". USENIX Standards. Retrieved April 2, 2004, from http://www.usenix.org/publications/login/standards/10.data.html

Mainelli, Tom. (July, 2003). "Are You Ready for a 64-Bit PC?" *PC World Magazine*. Retrieved April 2, 2004, from http://www.pcworld.com/news/article/0,aid,111508,00.asp

Mauro, Jim. (July 1998). "Asynchronous I/O and Large File Size Support in Solaris." *Sun World*. Retrieved April 2, 2004, from http://sunsite.uakom.sk/sunworldonline/swol-07-1998/swol-07-insidesolaris.html

Microsoft. (2004). "Windows XP 64 bit Edition Overview". http://www.microsoft.com/WindowsXP/64bit/evaluation/overview.asp

Open Group. (2004). "64-bit Programming Model: Why LP64?" Retrieved April 2, 2004, from http://www.unix-systems.org/version2/whatsnew/lp64_wp.html

Ostergaard, Jakob. (November, 2001). Retrieved April 2, 2004, from http://www.beowulf.org/pipermail/beowulf/2001-November/001935.html

SGI. (2004). "64-bit Linux". Retrieved April 2, 2004, from http://www.sgi.com/servers/altix/64bit_linux.html

Thanawala, Milan. (August, 2000). "Should I Move My Application to the Solaris Operating Environment with 64 Bits?" Retrieved April 2, 2004, from http://developers.sun.com/solaris/articles/64bit.html

Zedo, Dave. (2004). "Corps Battle Simulation". PEO-STRI web site. Retrieved April 2, 2004, from http://www.peostri.army.mil/PRODUCTS/CBS/