

FROM RAW DATA TO RUNNING SYSTEM: REDUCING MISSION REHEARSAL AND TRAINING APPLICATION PRODUCTION TIMELINES USING THE COMMON DATABASE (CDB)

Nick GIANNIAS
Presagis
Quebec, Canada

ABSTRACT

Traditional synthetic environment database generation techniques have favored proprietary formats closely tied to the client systems in use. This has primarily been a runtime performance issue, requiring a publishing step to assemble and optimize source data in keeping with the requirements of the client. The availability of high performance computers, especially multi-core processors, has brought into question the need for proprietary databases and the potential for using the additional computational capacity to standardize on a single database format. The CDB is a database specification designed to be a single format that can be used to serve a variety of clients as a runtime database. Through the use of runtime publishers, the clients can benefit from rapid updates to the CDB, thus reducing the overall time required to produce a training application or perform mission rehearsal. The individual elements of the database generation process along with the steps involved has been presented. Tests have been carried out to compare the time required to generate a CDB with other database formats. Results have shown that use of the CDB can reduce the overall production timeline when multiple clients are integrated as part of an overall system.

INTRODUCTION

Until recently, most databases were mainly processed using an offline approach and were customized to the specific runtime system with which they needed to work. The offline approach, while necessary because of the proprietary nature of the runtime system, was a daunting process for larger simulations, often requiring hours and even days of processing before the database could be published to the required format(s). Correlation errors resulting from multiple compilation/publishing tools were frequent, and updates or changes were often difficult and very time consuming to implement. This laborious process needed to be repeated for every runtime instance, including visual system, SAF, and sensor, as each runtime system required its own unique database. In the end,

compromises were made that reduced the level of portability, interoperability, scalability, abstraction, and correlation of the database with other simulator client-devices.

As a result of these database and database processing issues, applications became more difficult and more costly to create and maintain. And, as the end user's requirements have broadened and have become more demanding and specific, these issues have become exacerbated. Most database formats in use today still require a full offline re-compilation of the database specific to each runtime client.

Advances in database tool capabilities have partially resolved some of the issues associated with the database production process by allowing a single tool to output the multiple databases from a single set of data. However, databases still remain tied to specific runtime systems and, since implementations across different suppliers are unique, databases cannot be shared. As a result, systems integrators are required to undertake redundant engineering efforts to support duplicate proprietary technologies. This makes the content creation pipeline unnecessarily complex and users need to be trained in the black art of optimizing the output of database tools and converters for the target runtime system.

The use of proprietary runtime formats also results in additional long-term costs as new systems come online, each requiring their own version of a database. This is even the case with systems from the same vendor since the runtime format may not be backward compatible. Even if the format was backward compatible, elements of the original source data were probably dropped to meet the real time performance requirements of the original system. With a new, higher performance system, some of those restrictions could be relaxed and more source data (eg. higher resolution, greater feature density, etc) could be used. However, because of the tight coupling between database and runtime system, the database would have to be regenerated from source data.

The UK MoD has experienced virtually all these issues as various systems have come online over the course of time.

The resulting duplication of databases is depicted in figure 1. It is interesting to note that at least eight versions of the UK exist and Salisbury Plain, a military training area, is said to have over thirty-five versions. The ability to rapidly update a database and make it “runtime ready” to satisfy a training or mission rehearsal objective is severely impacted by the need to support so many database formats.

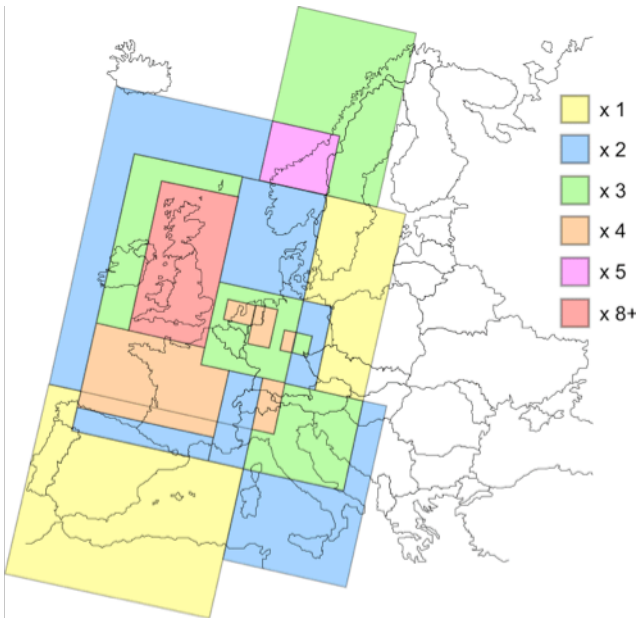


Figure 1: Database duplication within the UK MoD. 79% of all areas and 41% of airfields are duplicated at least once [1].

The need to mitigate the effect that multiple, heterogeneous, runtime clients have on the time required to field new databases led the United States Special Operations Command (USSOCOM) to develop the Common Database (CDB) specification, which is now an open commercial standard [2]. As the name implies, the intent of the CDB was to produce a single version of a database that can be shared by all simulation clients at runtime, thus removing the time-consuming offline publishing requirement to individual proprietary formats. This is accomplished through the use of a runtime publishing (RTP) component that is unique to each client system. The process of creating an RTP is beyond the scope of this paper; reference 3 provides an excellent overview of the experiences of one particular vendor.

BACKGROUND

To understand the existing typical production timeline, it is best to begin with an understanding of the various elements involved in the content generation process, illustrated in figure 2.

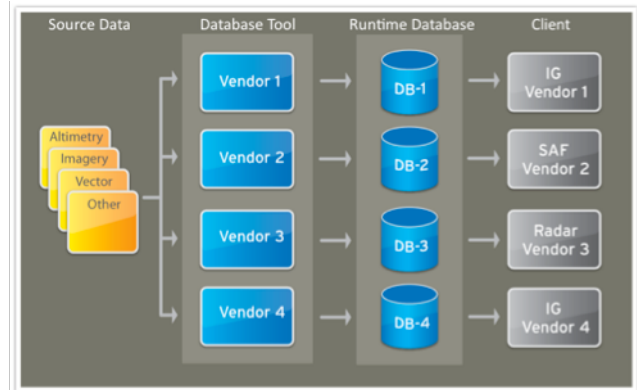


Figure 2: Elements of the existing content generation process

The first step element in the process is *source data*. Geographic areas of interest are identified and source data in the form of imagery, elevation and a variety of vector data representing lineal, areal and point features is gathered. Source data can also be in the form of interchange formats such as SEDRIS, or de facto standards like OpenFlight.

Source data usually requires some data preparation and modeling tasks, similar to those applied to data intended for use by GIS applications, such as formatting, error handling, geo-referencing data, registering data layers with one another and data harmonization. However, simulation applications usually require additional steps, namely data intensification and/or data decimation. The result of all this work is refined source data.

The next element in the process is a *database tool*. It takes as input the refined source data and generates as output the third element of the process, the *runtime database*. Sometimes referred to as database compilation (modeling and simulation terminology) or packing (game development terminology), this step transforms the refined source into a form closely aligned to the constraints, structure and formats of each of the simulation devices. The result is one or more databases in a proprietary format that can only be ingested by the appropriate *client*, the final element of the process.

Aside from being time consuming, this process has three major limitations:

- i. Proprietary databases require proprietary tools that may process the source data differently, resulting in

IMAGE 2009 Conference

- correlation errors
- ii. Proprietary databases cannot be reused between clients, resulting in unnecessary duplication of data and greater management overhead to track and synchronize versions
- iii. Database content is highly dependent on client performance and therefore databases are created based on client limitations instead of the availability of source data

The first limitation can be addressed by using a common tool to manage and process the source data. The second and third require a fundamental shift away from traditional database concepts.

The use of a common database tool in the content generation process is depicted in figure 3. Source data is imported into the tool, in this case Terra Vista, and through the use of *output compilers*, multiple proprietary databases are created. Output compilers are the link

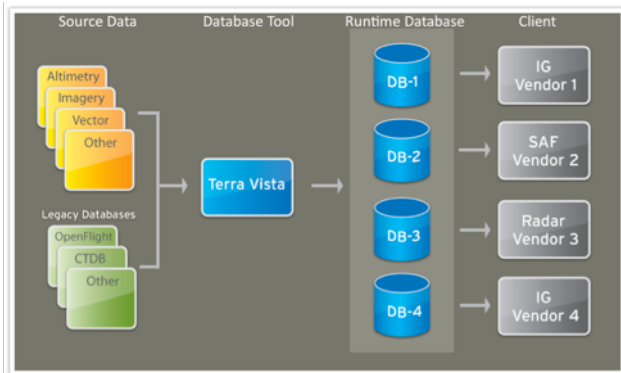


Figure 3: Elements of the content generation process using a single tool.

between a common environment representation (CER) within Terra Vista and the specific format requirements and restrictions of the client runtime system. Output compilers use the Terra Vista API to extract information from the CER and publish the client database¹. The CER can be thought of as an internal scenegraph that has been generated from the combination of source data imported into Terra Vista, the processing rules that have been applied to that data (e.g. elevation filtering) and the culture generators used to enhance the content (e.g. scattering trees). As a single tool applying the identical algorithms,

¹ While Terra Vista provides a range of output compilers as part of the product, the API is designed so that clients can develop their own output compilers as plug-ins to Terra Vista, thus enabling the generation of even proprietary formats through a single tool.

the chances of introducing correlation errors at this phase are less than with multiple, independent tools. An additional benefit to using a single tool is the simplified workflow; all source data and processing parameters can be managed in one location.

A single tool cannot resolve all potential correlation issues however when running clients from multiple vendors. The publishing step, which must account for the specific formats and limitations of the clients, is where miscorrelation can still be introduced. This is particularly true when the environment representation is fundamentally different. A typical example is the representation of terrain elevation as either a rectangular grid of elevation posts or a triangular irregular network (TIN). These two representations cannot be explicitly correlated and instead must be implicitly correlated by ensuring that any elevation mismatch between the two falls within an acceptable tolerance for the intended application.

The challenge of correlating different clients goes far beyond simple data representations. Parametric modeling is increasingly being used, either as part of the database production process as implemented by Terra Vista and other tools or as part of the runtime client as implemented by many games. Correlation in this case is very difficult without detailed knowledge of the algorithms and parameters being used. This topic is beyond the scope of this paper and the reader is referred to [4] for a discussion of the challenges of parametric modeling.

Thinking back to the problem the UK MoD faces with its myriad of non-reusable databases and defining a process that instead maximizes reuse, we come up with an idealized version that creates a single database that can be used by all clients, regardless of their vendor origin or purpose. Furthermore, the limitations of client performance, instead of being handled through the management of database content, are instead relegated to the client, thus decoupling the construction of the database from its use. This idealized process is depicted in figure 4.

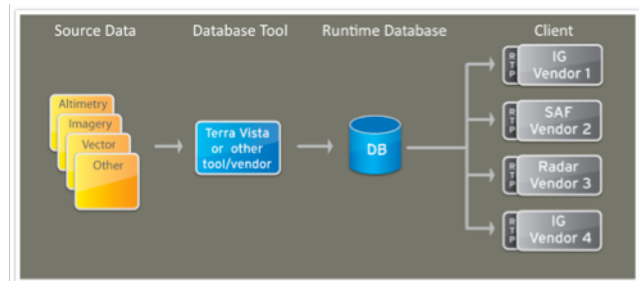


Figure 4: Elements of the content generation process using CDB.

IMAGE 2009 Conference

In order to fully realize this process, all clients must agree on both the format of the database and the conventions used therein. Considering how difficult it would be to get agreement from all vendors on a single runtime format, the CDB took the approach of defining a database format using already existing, widely accepted, industry formats and encouraging the development of format translators during runtime, referred to as *runtime publishers* or RTPs. The RTPs work as a front-end process to the clients, reading the CDB and generating optimal in memory representations that are unique to each client. This process entails additional computational and memory requirements but the payoff to the end user is enormous: a client with an RTP can “play” a CDB as-is, without compilation and without modification. Furthermore, because the CDB defines a robust set of conventions for everything from 3D models to material codes to vector data, all clients can immediately benefit from a richly attributed synthetic environment. Conventions are key to understanding the potential of CDB as a refined source data repository, in addition to its use as a runtime database.

Removing the need for a database tool by reading a variety of data formats directly for visualization or simulation is not enough. That data needs to be properly attributed in order for it to be useful; e.g. shapefiles without conventions are simply storage containers that can contain virtually anything. At some point in the process, there must be knowledge of these conventions, encoded in the original generation of the data (using a tool, just like before) or in the format converter of the visualization or simulation system as in figure 2 (another tool), or as a runtime process. In the last option, a different runtime software module would be required for every set of format/conventions but a tool would still be required to create those formats and conventions in the first place. The CDB thus represents the best option for reusability (because of the conventions it defines), openness (because it’s a specification and not software) and ease of implementation (because it uses the most common data formats instead of defining new ones).

PRODUCTION TIMELINE

The content generation process is but one aspect of the entire production timeline. Testing and deployment are also important and may have a significant impact if multiple databases have been generated and each needs to be copied to a client device. The major steps involved in production are: data preparation, data enhancement, data processing, deployment and finally testing. Each of these

steps is described below:

Data preparation: begins with the assembly of source data ensuring proper coverage, analyzing existing attribution and assessing quality. This process often includes the use of imagery processing tools to color-balance, geo-reference and perform classification. For vector data, this would include the use of GIS tools to eliminate or rationalize duplicate or overlapping data and filling in missing attribution. A quick-look of elevation data would also be performed to identify missing or invalid data. For 3D models, this step usually involves cataloguing existing models and identifying what new ones need to be created.

Data enhancement: this step can involve a combination of manual and automated tasks. Manual tasks include the creation of 3D models and the further attribution of vector data. Automated tasks are based on the creation and configuration of rules that can, for example, be applied to the extrusion of 3D buildings from footprints, trees over forested areas and power poles along electrical transmission lines. Enhancement includes the generation of material data from vector sources, for example a water material texture from an areal defining a lake and the creation of micro-textures to improve visual cues. Various other forms of data enhancement include smoothing of vector data (e.g. splining), altimetry blending and feathering, image mosaicing. There is no clear distinction between the data preparation and data enhancement phases; it depends mostly on the tools and processes that are used.

Data processing: this step involves the actual generation of the runtime database for client devices. Since most formats are unique to the client systems, either detailed knowledge of the format is required or a conversion tool supplied by the client system vendor. Depending on the tool and format, this step may require a complete regeneration of a database for even a small change.

Deployment: this is the copying or distribution by other means of the output database(s) to the individual client systems.

Testing: testing is usually broken up into two phases – verification and validation. Verification tests are performed against the specification (e.g. “is the database coverage correct”, “are all the specified features modeled”) and validation tests are performed against the requirements (e.g. “can you perform a visual approach”). While some verification tests can be conducted on a

IMAGE 2009 Conference

development workstation, validation tests are nearly always performed on the client system.

The CDB has several properties that have the potential to reduce the production timeline for both initial generation of the database and subsequent updates that are often required as part of the testing process. Namely, the CDB:

- i. contains data for all clients thus requiring only one database to be produced
- ii. structures data as multiple independent layers allowing for incremental updates of individual layers without affecting the others
- iii. structures data in a format that allows for efficient real-time access allowing it to be used as *the* client runtime database

In the next section, we will compare the production timeline using CDB with other data formats.

RESULTS

Database generation to CDB format is a new addition to Terra Vista version 6. The product was scheduled for release after the publishing deadline for this paper requiring that an engineering build be used for testing. The reader is therefore cautioned that the results could change upon release of the final product, our experience being that performance will increase, i.e. processing time will decrease.

All database generation tests used an approximately 0.25 x 0.25 deg geocell area centered around Camp Pendleton in southern California. The database contained multiple, high-detail areas with sub-meter imagery, DTED level 2 or finer elevation and densely populated culture, primarily trees and buildings. The database also contained a polygonal inset, in accordance with the CDB specification, representing an airport. The tests were conducted on a laptop computer and Terra Vista's multi-machine processing capability (MMB) was not used. Since the purpose of the testing was to compare relative performance, no attempt was made to optimize to reduce the total time of the database build.

Terra Vista's variable level of detail (LOD) capability was set up to process the database in accordance with the resolution of the source data. CDB has predefined levels of detail and Terra Vista automatically creates those without the need for extra setup.

The first set of tests compared the time to output a CDB,

VSB (Vega Prime binary format), CTDB (Compact Terrain Database – JSAF) and FLT (OpenFlight). VSB is primarily a visual format and CTDB a semi-automated forces (SAF) format. OpenFlight is often used as an output for interoperability since it is such a widely adopted open commercial standard. The results are shown in figure 5. CDB required the most time to process, followed by

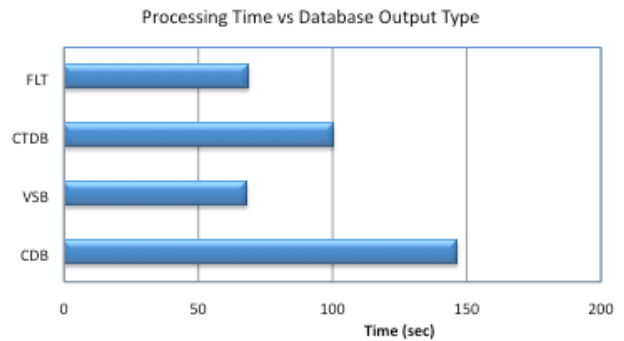


Figure 5: Processing time for four different database output types.

CTDB and then FLT and VSB. This is not surprising since CDB contains a superset of data included in VSB and CTDB. CTDB processing time is lower because it does not require imagery, which can be very time consuming to process. The results of other visual formats are likely to be very similar to those of VSB, which is representative of a traditional visual database generation process that creates an optimized, proprietary database. In an environment that integrates a visualization and SAF, both databases would need to be produced and the total time would be greater than producing either database alone, although less than the sum². The advantage of CDB in an environment that integrates multiple systems is in the elimination of the requirement to produce multiple databases.

The size of the resulting databases is presented in figure 6. As expected, the highly compacted, client specific formats CTDB and VSB are the smallest. Next larger is CDB, due in large part on its reliance on source formats (Shape, Geotiff, JPEG200, XML) and finally OpenFlight.

The second set of tests focused on the time to process the individual data layers of CDB. The separation of data into independent layers is one of the aspects of CDB that was designed to permit rapid updates; changes could be made

² The reason for this is that Terra Vista exacts an overhead associated with producing the CER that is only required once, even if multiple databases are being produced.

IMAGE 2009 Conference

to a single layer without affecting the other layers. Here we would expect the processing time to vary depending on the data layer being generated and in fact it does, as shown in figure 7. Culture and imagery take the longest time to process, followed by elevation and raster material. Imagery processing time is dependent on the resolution of the source data, which is sub-meter over a significant part of the database. The shorter processing time of the

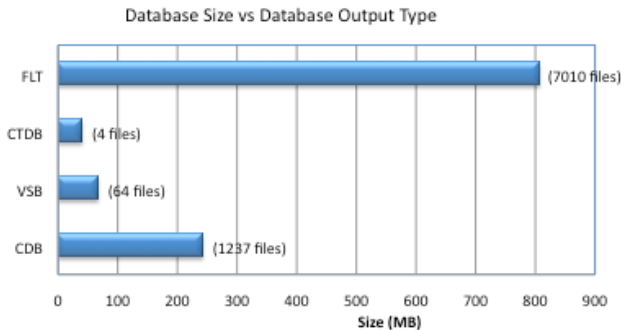


Figure 6: Database size for four different database output types.

elevation dataset can be attributed to its lower resolution (as compared to imagery) and the relative lack of sensor and material textures in the test database result in minimal raster material generation times. The processing time of culture data seemed unusually high and some additional testing was performed to characterize it. This involved

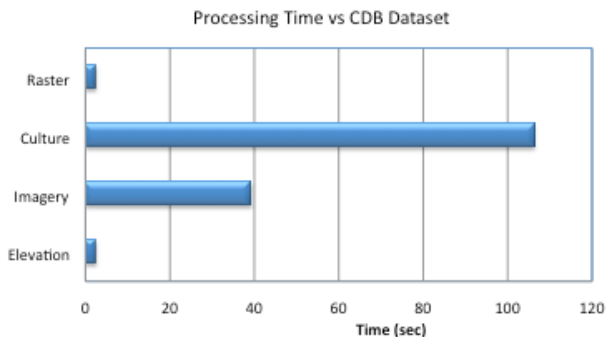


Figure 7: Comparison of database processing time by CDB dataset.

changing the number of blocks in the gaming area. Blocks in Terra Vista represent “processing chunks” and judicious selection of block size can improve processing times. As a baseline, all tests were performed using an 8 x 8 block gaming area. To see what impact block size would have on culture processing times, the block size was increased,

resulting in a 4 x 4 block gaming area. The effect was dramatic, cutting processing times by 50%. The same was not true however with the other CDB datasets where the effect was less than 10%. Clearly, a method that would assist the user to select the optimum block size for processing, possibly by dataset, would be a welcome addition to the tool.

Figure 8 compares the resulting size of the individual CDB datasets. Not surprisingly, imagery represents the majority of the CDB overall size, followed by elevation. The majority of the culture content size is attributable to the geometry and texture files of the OpenFlight models. In this database, only the airport area contains a raster material dataset to represent the runway, taxiway, etc material information and therefore has a minimal impact on the overall size. If raster materials had been applied to the entire database, say for example for sensor simulation, this dataset would have been significantly larger.

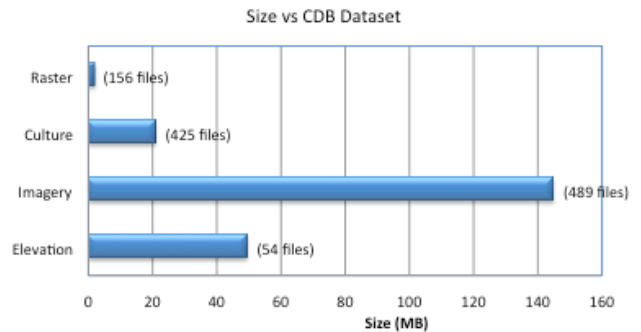


Figure 8: Comparison of CDB dataset size.

Looking at these results in light of the production timeline, we can analyze the impact of CDB.

Data preparation: CDB is unlikely to have a direct impact on the data preparation phase which is primarily a data collection and cataloging process.

Data enhancement: CDB can benefit the data enhancement phase by speeding up the process of reviewing the effect of changes to the data in each client system. There are two main contributors to this:

- The results show that it is quicker to produce a single CDB than multiple databases, one for each client
- Changes to CDB can be performed on an

IMAGE 2009 Conference

individual data layer without the need to regenerate the other data layers. The results show that the processing time depends on the data layer which saves significant time in not having to process other layers.

Data processing: as stated above, the results show that it's quicker to produce a single CDB than multiple databases. The fact that the same CDB can then be used by multiple clients is a net reduction in the overall timeline.

Deployment: no specific testing was carried out to compare the deployment of a CDB with that of multiple independent databases. One of the criticisms leveled against CDB is its size and number of files. The author has experimented with both of these aspects and found that the size of a CDB is mitigated as a result of JPEG2000 compression applied to imagery and LZW compression applied to elevation and raster material. These three datasets comprise the vast majority of the size of the CDB. The large number of files comprising a CDB, while an essential element of its open architecture that eschews proprietary database engines and relies entirely on existing data formats, can create problems when deploying or copying a CDB. This author's experience with a 700+ GB CDB is indicative of the challenges associated with applying existing processes to huge databases. For example, every attempt to copy this database failed using Microsoft Windows XP. A variety of Microsoft and 3rd party copying programs were used and all generated errors by Windows. Research on Microsoft's support database revealed a known problem with Windows XP when backing up large volumes [5]; installing Windows XP Service Pack 3 resolved the issue. A second problem resulting from the large number of files is the overhead associated with creating those files when copying from one device to another. The time to copy the 700 GB database from an internal drive to an external drive connected via USB2 was estimated to be 15 days. Using the identical hardware except this time the external disk was connected using an eSATA interface, the entire copy successfully completed in 21 hours. This is comparable to performing a sector-by-sector copy which is the fastest possible method, assuming the user has the option to reformat the target disk. The lesson learned is that while CDB allows you to build huge databases, managing that amount of data is a challenge that is not unique to CDB.

Testing: no specific analysis was performed on the impact of CDB on the testing phase. This is one area that could benefit greatly from rapid turnaround, as supported by

CDB and potentially even automated verification testing applied to individual CDB datasets.

CONCLUSIONS

The early experiences of producing CDB have highlighted the potential of this database specification and associated tools to reduce timelines associated with simulation and visualization applications. The test results, while promising, still need to be validated with actual user applications. Processes that have been in use for many years benefit greatly from the lessons learned by a generation of practitioners; only through widespread use and sharing of experiences can CDB fulfill the vision of a single synthetic environment database for the modeling and simulation community.

ACKNOWLEDGMENTS

The author would like to thank Joe Burns, Product Director and Subject Matter Expert of Content Creation at Presagis for his encyclopedic knowledge of Terra Vista and database formats.

REFERENCES

- [1] Fawkes, A., 2007 "UK Simulation Developments", *Presentation at the International Week of Simulations*, Ljubljana, Slovenia.
- [2] *Common Database (CDB) Specification*, Presagis, Montreal, Canada.
- [3] Ford, B., Nigus, S., Saute, B., 2008, "CDB Real-Time Publishing – The Transition", *Proceedings of the 2008 IMAGE Conference*, 95-103. St. Louis, Missouri: The IMAGE Society.
- [4] Brockway, Dan E., 2004, "Database Correlation in an Increasingly Parametric World", *Interservice/Industry Training, Simulation, and Education Conference*, paper no. 1824, Orlando, Florida.
- [5] <http://support.microsoft.com/default.aspx/kb/304101>