W - O - W
Data Warehouse, SAS/OR and the WEB
A Decision Support System (DSS), using a Data Warehouse designed for Project Management
and optimized for SAS/OR, Distributed on the WWW.

Art Rerecich
LOCKHEED MARTIN
Space Launch Decision Support Systems, Lead
arthur.m.rerecich@lmco.com

ABSTRACT

Although our operational systems are very adept at processing transactions they don't lend themselves to ad-hoc reportage or analysis. Operational and senior management need distilled information from these systems for effective decision making. We were tasked to deploy a Decision Support System for executive level program surveillance and On-Line Analytical Processing (OLAP).

We designed a Data Warehouse with the SAS system, running in a Client/Server configuration (UNIX & Windows 95), SAS/OR, with output distributed across the World Wide Web (WWW), to accomplish this task.

INTRODUCTION / BACKGROUND

The Space Launch Systems group of Lockheed Martin's Astronautics (Denver) Division is responsible for the design, fabrication and delivery of two major space transportation systems; the Titan and Atlas missiles. Each of these programs are sub-divided into two additional programs, lower stage (Titan Core or Atlas Booster) and upper stage (Titan or Atlas Centaur).

Although they share common operational systems (central functions) these four programs have different management, contracts and customers. We were initially funded by the Titan 4 Core (lower stage) program. Their job is build and deliver the vehicle to the launch site. This involves the coordination of several disciplines: engineering, procurement, manufacturing, test, etc.

Program management needed a cost effective way to monitor the performance of their operational systems and insure they were working in concert. Some of these activities are in different operational systems and environments. Therefore, manual processes were used to create integrated views for upper management and our customers.

As you might imagine, with a program this large and complex, by the time all of the information was updated, it was nearly out of date. Our goal was to streamline the data collection process, transform the operational data for consistency and make the data, as well as the analysis produced, readily available.

BEFORE WE BEGIN

This paper is a three part description of our system:
1. Describe the business criteria and technical requirements we needed to satisfy.
2. Explain the use of SAS/OR in our Data Warehouse environment.
3. Give an overview of our distribution system using the Web.

However, there are a few high level items or *lessons learned* that I'd like to mention which were critical to our success. I call this my movie trivia "Fearsome Foursome".

-1-
"We're on a mission…"
The Blues Brothers
- Get a clear idea of what your goal is and stick to it. This will become your compass. As your project evolves you will lose your way if this is not firmly established, not only to you, but your team and management sponsor as well.

- Analysis systems are fundamentally different from operational systems. Although this may seem obvious, I will guarantee to you it is not universally perceived as such. An analysis/data warehouse will be difficult to justify, develop and manage if it is viewed as an extension of the operational environment.
- It is not enough to have a 'missionary zeal' about your project, you must be as well read as possible. Bill Inmon has over 30 books on Database Systems & the Data Warehouse. There are numerous other sources as well, make use of as many as necessary.

-2-
"Deserves got nothing to do with it…"
The Unforgiven
or
"Show me the money…"
Jerry Maguire

- Your management sponsor must be committed and **FUNDED**. Remember you are proposing something to a company that hasn't needed it before. Your challenge is to convince them that they need it now and this can be a hard sell.
- If your management sponsor believes in the project, they can do a lot of selling for you. They know the business, they speak the language, they know what they need, and consequently they will be more believable to upper management.

-3-
"If you build it they will come…"
Field of Dreams

- If possible start small and build out, a classic spiral approach. Think of a retail chain's approach (Wal-Mart, Target, etc.). Build one store, make it successful, then explore another market.
- We were initially funded by the Titan Core program. Although we warehoused operational data for all programs, we provided products only to one. As we became more visible, other program managers, seeing our products, soon came to us for support. This creates an important dynamic; they are **BUYING** from us instead of us **SELLING** to them.
- Use this same approach for your development and design. We employed a Spiral / Rapid analysis and design approach. The basic nature of analysis systems is their volatility. If you think back to the retail model, when you open a store, you don't need to know what the entire country's needs, just that area.

Consequently don't try to understand the entire company or warehouse all of their data, just what your sponsor needs. Get just enough data or requirements to develop, then produce. Once something is available review with your customer, get more (information, data, etc.) and produce again.

This is not something you do for a while or a front end process, rather we've found that it is our normal operational environment. Our original management sponsor has never "finished" his requirements definition, consequently my staff assigned to him never becomes available for another assignment.

- Since we don't plan on finishing our requirements definition, when other programs come to us we scope additional staff to support them. Obviously not all additional work is created equal, sometimes it's less than a full person, sometimes more. However, the more times we complete the analysis/development cycle we become better at estimating resources and it enables us to "Grow Smart". Also, since they are coming to us for our "proven" product, we're in a better position to establish requirements and justify additional resources.

-4-
"A man's got to know his limitations…"
Dirty Harry III

- It is critical to understand your environment's strengths and weaknesses to adequately manage you customer's expectations. Remember the difference between expectation and reality is anger.
- For example in our environment it is very expensive to put SAS on the desktop (~$3500/seat). Consequently we've found the Web an extremely viable alternative for data dissemination. However, as wonderful as the Web is, it can't do everything. Since we have a limited number of SAS seats available we must be careful how we allocate them.
- You must know your environment from both a technical, what it can do, and performance, how well does it do it, perspective. With analysis systems in general and the Web in particular, there are no standard benchmarks from which to judge. We monitor our system from many perspectives to get as clear an idea of what's going on as possible.

Part One
BUSINESS / TECHNICAL SUCCESS CRITERIA & FUNCTIONAL / TECHNICAL REQUIREMENTS

A project like this exists for the simple reason that there is a well defined business need with an information based solution. We will discuss the two major parts of this process, the success criteria and functional requirement that result.

Business Success Criteria:

As you might imagine the most common reason to spend company money on the business process is to make is perform *Better, Faster, Cheaper*. Consequently our business success criteria follows this overall approach in three ways.

- **Strengthen Cost Effectiveness-** We need to respond more quickly to design changes, procurement and production requirements, as well as monitor modification kit, milestone availability and delivery schedules.
- **Improve Schedule Dependability-** Our implementation plans must be cross functional, visible and viable. We need to improve not only how they are produced, but maintained as well. We also must insure synergy among all functional elements.
- **Enhance Operability-** The output products must come from integrated operational data and be available to users with minimal effort. The system must conform to our division's overall information architecture. Our goal is for *Horizontal Data Distribution* for all production information to verify contract/design configuration and current production or procurement status, across all functional areas and integrated product teams, while maintaing *Vertical Data Integration* with the existing information architecture.

Technical Success Criteria:

Our general goal in this area was to move forward and not leave anything behind. Consequently, our technical approach is also in three parts.

- **Leverage Existing Legacy Activities-** Part of moving forward is to replace the existing environment without losing any products or functionality.

- **Evolve the Environment-** Bring the information closer to the end user with full support for client-server and Web technologies. The system must also be able to scale, grow or shrink, consistent with our business needs.
- **Synergize with the Company's Information Architecture-** Again, it can't be stated often enough, the system must 'fit in' with our technical environment. Our corporate goals are commonality and consistency. The system must run on common platforms and support industry standard databases and protocols.

Functional Requirements:

Most systems function in three general parts, data must come into the system, it must be processed in some way and the results made available to the users.

- **Input-** The system must be able to access data from multiple sources, environments and platforms. This access must be functional in both batch and interactive environments.
- **Processing-** The system must have a robust language for data set manipulation and application development, which must be cross platform deployable. It must also satisfy standard program planning requirements, consisting of, but not limited to the following:
  - ⇒ aggregate across time horizons,
  - ⇒ support flexible schedule options,
  - ⇒ support custom calendars and 'What If' analysis.
- **Output-** The system must be able to generate the standard mix of business and program planning information products; gantt charts, networks, bar, line and area graphs, etc.

Technical Requirements:

Simply put the system must be flexible. A lot of what we do is 'evolve' legacy (proprietary) systems toward a more open, industry standard environment. Our technical requirements reflect these goals.

- **Presentation Layer - PORTABLE**   The system must be deployable on all of our standard platforms (PC, Mac, UNIX). It must also be able to offload processing to the client.
- **Application Layer - EXTENSIBLE**   The development / application environment must be able to integrate external (C/C++, Operating System) routines into existing processing.

- **Database Layer - SCALEABLE** The database must be able to grow or shrink as required by our business. We not only have to accommodate growth but downsizing and insulated (special) programs as well. It must also be accessible from third party data access tools (ODBC, Web, etc.).

Part Two
DATA WAREHOUSE DESIGN & PROJECT MANAGEMENT (SAS/OR) CONSIDERATIONS

Our primary customers (initially) were executive level decision makers, therefore we based our design approach the following:

"Storage is Cheap, Time is Expensive"

Put another way we were less concerned with the amount of tables or the degree of de-normalization as we were with the speed of execution. Consequently, we chose a Categorization Design Methodology.

Also, since our customers were Project Managers the warehouse had to support their decision making process. Therefore, it was critical that it work with the Project Management features (procedures) of SAS/OR (CPM, GANTT, NETDRAW).

Categorization Design Methodology:

Our starting point was data coming from normalized, transaction processing systems, available to us in several formats (text, Sybase, etc.). Our task was divided into two parts; get the data of interest into SAS, then de-normalize to make it readily available to our customers.

The normalized operational data comprised the atomic level of our environment. Although our entire warehouse was fairly small (2-3 million records, less than 10Gb) we wanted to use derived (categorized) data wherever possible. Here's an overview of the process:

- Denormal operational data based on subject categories.
- A single processed join occurs during weekend processing (~30-40 minutes).
- Categorized output satisfies 80%-90% of our reportage.
- This approach eliminates the need to continually re-join or create logical tables.

- We will continue to review Star & Snowflake schema methodologies as capacity increases and these technologies evolve.

Project Management (SAS/OR) Considerations:

The nature of our business is that we make very few products for very few customers. However, the products that we produce are expensive, comprised of many parts (~10,000 per vehicle), have deep indenture structures (~20 levels) and have long, complex fabrication cycles. Most operational requirements systems (MRP) are designed for high volume, short execution cycle products. Consequently, our operational system doesn't do any re-scheduling after an order is firmed. It keeps track of it's relative status (firm, picked, released, complete) however the dates must be manually changed.

Our customer's questions generally focus on cost, resource management and schedule in the following areas:
1. Where are you in the build cycle?
2. When will you be done?
3. Do you have a good plan to get there?
4. WHAT IF we changed something?

When we did our trade study, one of our most critical requirements was the ability to schedule and analyze large data sets (40,000 to 250,000 records). The capability of SAS/OR was one of the primary reasons we chose SAS for our project.

Although the source data comes from a requirements system, we perform two major processes on the data to satisfy our requirements:
1. First the categorization and de-normalization process creates a good set of open requirements, driving from the lowest child to the highest parent.
2. Next we pick up additional descriptive information as required so everything is in one place, the "ALL" dataset.

3. We then sub-divide to two levels of program specific information, unique (parent/child) build and buy items. For example for the Titan 4 (T4) program the sequence will look like this:
    a) ALL (290,000 records),
        i) T4ALL (58,000)
            a) T4Make (24,000)
            b) T4Buy (6,900)
*NOTE: T4Buy is a unique subset of BUY orders. Each BUY order is listed once even though it may support many MAKE orders (parents).

4. This approach satisfies a lot of our general purpose business reportage. We'll then further extract from the MAKE dataset the level of orders that most affect the program, which we classify as 'log' level, meaning assigned to a specific vehicle. The result is the T4Log dataset (4,000 records).

5. Next we'll process this dataset (T4Log) to insure a complete structure, from top (children with no parents) to bottom (parents with no children).

6. Finally we'll run the CPM procedure against this dataset and generate various schedule and resource charts.

Part Three
WEB DISTRIBUTION SYSTEM

The final part of our system was making all of this wonderful information available to our customers. When we first began this project last year, long ago in "Web Time", we believed that the Web would be just one part of our distribution strategy.

Since it was too expensive to put SAS on every desktop, we originally planned a mixed distribution environment consisting of open Client/Server access tools (Powerbuilder, Omnis 7, etc.), ODBC applications (Excel, GQL, etc.), SAS clients and the Web. As our project and the Web evolved, our motto became:

"THE WEB IS EVERYTHING,
EVERYTHING IS THE WEB."

Although there will be times when we'll need to deploy the full capability of SAS to a user's desktop, for the vast majority of our users, Web access is sufficient. Our Web based distribution system can best be described in two parts; development environment and page structure.

Web Development Environment:
    The development tools available also break down into two areas:
1. **Industry Tools-** These are the general purpose, industry tools for server configuration, page development and management.
2. **SAS Tools-** These are the specific SAS procedures, macros and drivers used to access data in SAS datasets.

Industry Web Development Tools
1. Authoring
    a) Text- Basic editors: vi (UNIX), Notepad or Wordpad (Windows)
2. Graphics
    a) Charts- GIF (SAS Image GIF Drivers), PDF (Acrobat PDF Writer)
    b) Image- Adobe Photoshop 4.0 & Corel Photo Paint
3. Web Browsing
    a) Netscape Navigator 3.0 & Microsoft Internet Explorer
    b) Plug-ins (Adobe Acrobat Reader 3.0)
    c) Helper Applications (M/S Excel & PowerPoint)
4. Server
    a) NCSA HTTPd Server 1.5.1 - Beta on UNIX
    b) Considering- Netscape Enterprise on UNIX & Microsoft Internet Information Server (IIS) on NT
5. Administration
    a) Manual- Considering Microsoft Front Page or Netscape
6. Security
    a) NCSA support scripts
    b) Considering Netscape Certificate Server
7. SAS Database Access
    a) CGI Scripting, Perl 5.002 & SAS 6.12

SAS Web Tools
1. Publishing SAS output on the Web
   a) HTML Formatting
   b) IMGGIF Drivers
   c) GIF Animation Driver
   d) IMGJPEG Driver
2. Interactively querying SAS data from a Web Browser
   a) SAS JDBC Driver
   b) HTMSQL
   c) SAS ODBC Driver
3. Dynamically running SAS programs and analysis
   a) SAS Automation Plug-in
   b) CGI Scripting

Web Page Structure:
Our Web pages are generally classified by how much work they perform. They can range from simply pointing to an image, to initiating a SAS session and running a program. Output can either be screen displays (graphics or reports), text or other program files (Excel, Adobe Acrobat, etc.).

There are four classifications or types of Web pages in our environment; Static, Dynamic, Interactive and Hybrid (Dynamic/Interactive).

1. **Static-** These are simple HTML pages with a link to a pre-defined output (file). The target file can be graphic or text in any browser supportable format (gif, jpeg, Excel, text, etc.). Depending on the file type the browser, if configured correctly, may have to launch a helper application (Excel) or plug-in (Adobe Acrobat Reader) to display the output properly.
2. **Dynamic-** These pages access data from SAS datasets, but don't initiate a separate SAS Session. They operate similar to or in conjunction with SAS/Share (soon to be SAS/Intrnet). They consist of a HTML form with dynamic objects that build things "on the fly". This can be HTMSQL queries, that drive drop down boxes or output. Although data forms are built in advance, there is no hard coding of data in them.
3. **Interactive-** Here we'll take data from an HTML form, parse it in a CGI script and initiate a SAS session on the host. The SAS session will run whatever programs or statements it needs and then return the results. Again the output can be in any Web supportable form (gif image, text file, Excel, etc.). We've been able to leverage the SAS/OR capability with this approach.

4. **Hybrid (Dynamic/Interactive)-** As the name implies these pages will do a little of both. They may use dynamic objects to develop select criteria as a front end to a CGI script that initiates a SAS session. We initially built reference tables for our drop down boxes, then went to HTMSQL. Each approach has it's strengths and weaknesses. Realizing that "one size does not fit all", we now strive to use the most appropriate approach.

General WEB Notes:
Working with the Web for the last year has been a real education. It is a distinct approach to information delivery that takes a little getting used to.

The main thing to remember is that a Web Page is virtual page and is not like a real page of paper (8.5"x11"). A Web page is designed for electronic display on a variety of monitors. Consequently, printing and image manipulation are real issues.

Here is a list of things to consider:
1. **Graphic Images-**
   a) Business graphics don't display or scale very well. SAS 6.12 came with multiple gif drivers of various sizes, you'll have to play with these to determine which work best in your environment.
   b) There are always trade offs with images related to size, type and quality.
      i) High detail images result in large file size and longer downloads (page performance).
      ii) Small size images display well on many monitors however don't print or import into other programs (PowerPoint) very well.
      iii) Full size images import and print well but smaller, lower resolution monitors have to pan a lot.

2. **Adobe Acrobat Writer & Reader-**
   a) This product is crucial to our page. It mitigates a lot of the limitations of the Web and there are certain things that just won't display properly any other way.
   b) However SAS doesn't provide a driver to create Acrobat (PDF) files and the version of Postscript that SAS uses is very old. Consequently the Acrobat Distiller, which converts Postscript to Acrobat (PDF) doesn't work very well.
   c) Since SAS doesn't provide a driver or conform to the latest Postscript revision, all Acrobat images have to be developed interactively.

3. **Excel-**
   a) SAS does not support their Excel output procedure (PROC DBLOAD) on UNIX. and since our Web and SAS servers are UNIX, we can't generate Excel files this way.
   b) There is a work around however. If your server has a MIME type defined for Excel, you can use a command in your 'Put' statement to generate Excel output. A sample line would look like this:

      PUT CONTENT-TYPE: APPLICATION/EXCEL

4. **Text Output-**
   a) Printing multi-page reports, with headers is very difficult if not impossible. Since one of the advantages of the Web is ease of access, we never know on what printer someone will be trying to print to or how they have their browser configured.
   b) Again, Adobe Acrobat can help minimize this issue.

SUMMARY

The rapid development of the Web as an information delivery system enabled us to create and deploy this system faster and more widely than we had originally imagined. The Web mitigates a lot of the installation, configuration and maintenance issues related to traditional Client/Server deployment.

We have found the Web an ideal vehicle for Executive level information delivery. Although the screens are bounded somewhat by our design considerations, they are very powerful and make a lot of information available in an easily maintained, robust environment. We've also found SAS on the desktop a good complement for managers (and others) that want a more interative approach to data discovery and analysis.

ACKNOWLEDGMENTS

A project like this doesn't happen without a lot of hard work from a number of people. Naturally my project team and management sponsors are the ones most directly responsible for our success. Their dedication and hard work really made the difference.

Additionally the SAS Institute and a number of their people contributed in very significant ways and that I'd like to acknowledge now:
1. The local SAS Office and the SAS Web page which offered a great deal of support, information and tons of code.
2. The SAS/OR group, especially:
   a) **Radhika Kulkarni, Ph.D.** who quickly responded to all of my questions and help me use SAS/OR in non-traditional ways.
   b) **Gehan Corea** who's excellent paper on multi-project scheduling from SUGI 21 became our blueprint for our What If scenarios.
   c) **George & Diane Tompkins** who's onsite training filled in a lot of gaps.

REFERENCES

SUGI Proceedings
- Corea, Gehan (1996) "Multiproject Scheduling using SAS/OR Software"
- Grippo, Karen & Chen, John (1996) "Building a Data Warehouse with SAS Software: A Recipe for Success"
  Proceedings of the Twenty-First Annual SAS Users Group International Conference

Web Server
- Magid, Matthews & Jones. *The Web Server Book*: *Tools & Techniques for Building Your Own Internet Information Site*, Ventana Press, 1995.
- Mudry, Jon. *Serving the Web*, The Coriolis Group Inc., 1995.

- Cravens, Dick & Budnik, Larry. *Offical Netscape SuiteSpot 3 Book*, Netscape Press, 1998

HTML
- Graham, Ian S. *The HTML Sourcebook,* John Wiley & Sons, Inc., 1996
- Oliver, Dick. *Teach Yourself HTML 3.2 in 24 Hours,* Sams.net, 1997

CGI and Perl
- Brenner & Aoki. *Introduction to CGI/Perl: Getting Started with Web Scripts,* M&T Books, 1996
- Deep & Holfeder. *Developing CGI Applications with Perl,* John Wiley & Sons, Inc., 1996
- Till, David. *Teach Yourself Perl 5 in 21 Days,* Sams Publishing, 1996
- Wall, Christiansen & Schwartz. *Programming Perl*, O'Reilly & Associates, Inc., 1996
- Gundavaran, Shishir. *CGI Programming on the World Wide Web*, O'Reilly & Associates, 1997

Java
- Boone, Barry. *Java Certification Exam Guide*, McGraw-Hill, 1997
- Deitel & Deitel. *How to Program Java,* Prentice Hall, 1997
- Heller, Phillip & Roberts, Simon. *Java 1.1 Developer's Handbook,* Sybex, 1997
- Horstmann, Cay S & Cornell, Gary. *Core Java 1.1 - Volume 1 Fundamentals*, Sun Microsystems/Prentice Hall, 1997
- Jamsa, Kris. *Java Now*, Jamsa Press, 1996
- Lalani & Jamsa. *Java Programmer's Library*, Jamsa Press, 1996
- Patel, Pratik & Moss, Karl. *JAVA Database Programming with JDBC,* The Coriolis Group, Inc., 1996

JavaScript
- Flanagan, David. *Javascript, The Definitive Guide*, O'Reilly & Associates, 1997
- Staner & Hail. *Netscape One Developer's Guide*, Sams Publishing, 1997

Graphics & Miscellaneous
- Deep & Holfeder. *Designing Interactive Documents with Acrobat Pro*, John Wiley & Sons, Inc., 1996
- Geiger, Kyle. *Inside ODBC, Microsoft Press*, 1995
- Inmon, William H. *Data Architecture: The Information Paradigm*, QED, 1992
- Inmon & Hackathorn. *Using the Data Warehouse*, John Wiley & Sons, Inc., 1994
- McCanna, Laurie. *Creating Great Web Graphics*, MIS Press, 1996