Fermi National Accelerator Laboratory

FERMILAB-Conf-91/173

Design of the Digital Sky Survey DA and Online System — A Case History in the Use of Computer Aided Tools for Data Acquisition System Design

D. Petravick, E. Berman, T. Nicinski, R. Rechenmacher,
G. Oleynik, R. Pordes, and C. Stoughton

Fermi National Accelerator Laboratory
P.O. Box 500, Batavia, Illinois 60510

June 1991

* Presented at the *7th Real Time Conference*, June 25-28, 1991,
Julich, Federal Republic of Germany.

Operated by Universities Research Association Inc. under contract with the United States Department of Energy

Design of the Digital Sky Survey DA and Online System – A Case History in the Use of Computer Aided Tools for Data Acquisition System Design*

Don Petravick, Eileen Berman, Tom Nicinski, Ron Rechenmacher, Gene Oleynik, Ruth Pordes, Chris Stoughton

Fermi National Accelerator Laboratory / P.O. Box 500 / Batavia, IL 60510 / U.S.A.

Abstract

As part of its expanding Astrophysics program, Fermilab is participating in the Digital Sky Survey (DSS). Fermilab is part of a collaboration involving University of Chicago, Princeton University, and the Institute of Advanced Studies (at Princeton). DSS main results will be a photometric imaging survey and a redshift survey of galaxies and color-selected quasars over π steradians of the Northern Galactic Cap.

This paper focuses on our use of Computer Aided Software Engineering (CASE) in specifying the data system for DSS. Extensions to "standard" methodologies were necessary to compensate for tool shortcomings and to improve communication amongst the collaboration members. One such important extension was the incorporation of CASE information into the specification document.

I. Introduction

The photometric imaging survey will be done in four colors over π steradians within the Northern Galactic Cap. 10^8 galaxies and 10^5 quasars will be imaged. From this pool of objects, 10^6 spectra will be obtained in a redshift survey of galaxies and color-selected quasars.

A dedicated 2.5m telescope and data acquisition system will be commissioned in January, 1994 at Apache Point Observatory, New Mexico. After a year of observing, during which the instrument will be calibrated and the implementation of data reduction completed, the survey will formally begin in January, 1995. The survey in the Northern Galactic Cap will take 5 years to complete.

The survey will be carried out using a special purpose 2.5m telescope with a wide (3°) field of view. Photometric imaging will be done with a 30-chip Charge Coupled Device (CCD) Camera, each chip being 2048^2 pixels in size. Spectroscopic data will be fiber-fed to two high-throughput, double spectrographs, each receiving light from over 300 fibers. Each fiber usually covers one astronomical object. The survey will also have an end-to-end data handling system that is capable of quickly generating object lists and an archive of all the original photometric images and spectroscopic data [1].

Fermilab is specifying the complete DSS data system, applying the expertise we have gained from constructing data systems for High Energy Physics (HEP) experiments. The DSS data acquisition system has characteristics similar to those found in HEP experiments. During photometric imaging, data will be acquired at a rate of 5 megabytes/second. In noncompressed form, a pixel-level map of the region surveyed will require over 12 terabytes of storage.

Two factors not present in HEP experiments affect our approach towards system engineering for the DSS data system:

- the small window of time during which observations can be done: approximately only 100 hours per year are suitable for photometric observations. The remaining 800 hours are suitable for spectroscopy. Since the detection of objects from the photometric images drives spectroscopic observations, photometry *must* be done whenever possible.
- the physical distance between observations and system expertise: the 2.5m telescope at Apache Point Observatory, New Mexico is over 1500 kilometers from Fermilab.

These factors have caused us to place greater emphasis on usability and reliability requirements. At Fermilab, support and training for the experiment's data systems are available locally. The accelerator runs according to a fixed schedule. However, DSS will run sporadically, interrupted by the positions of stars, weather, atmospheric conditions, and moonlight.

It is critically important that the data acquisition system be reliable, simple, and relatively easy to learn. Data acquisition will run for 6 years at Apache Point. The scientifically useful life of the data, after it is reduced and calibrated, is expected to be over 20 years.

The need to construct a data system with these attributes has led us to re-examine our engineering methodology. We have improved our abilities in stating usability requirements and simplifying the system. This has been done by adapting the systems engineering techniques

*Sponsored by DOE Contract DE-AC02-76CH03000.

learned while working on HEP experiments to this project in Experimental Astrophysics [2].

II. The System Specification

The DSS specification involves the use of Structured Analysis (S/A) methodologies [3] along with a specification document [4]. While S/A, done within the confines of a CASE tool, is an improvement over ad-hoc system specifications, the use of these tools alone does not provide a framework for a comprehensive set of specifications. For example, "standard" S/A methodologies do not address architectural and hardware considerations. Analysis of a system and its specification involves the simultaneous evaluation of system requirements, functional specification, and system architecture.

As no CASE tool incorporates the three previously stated needs, we have supplemented our S/A methodology with Architecture Diagrams [5], additional requirements, and notations about the model (described in the next section). We have also identified a set of references [5, 6] which clarify ambiguities found in literature describing S/A techniques.

We have developed a methodology which incorporates all of these aspects of the system into a specification document. It is no surprise that a draft specification contains errors; up to 25 major errors per page are not unusual [7]. If a major error passes from the specification stage to the design stage (or even further), it will require more time to fix later than if it were caught in a specification review. Thus, specifications must be "debugged," just as code, before they become useful.

Our method allows individual study of the specification apart from oral presentations. This allows the specification to be reviewed and incrementally refined by consensus of the collaborators.

III. Structured Analysis Extensions

Architecture Diagrams

Data Flow Diagrams (DFDs) are used as an organizational focus for the construction of the specification document and other aspects of our S/A. After a first cut DFD is obtained, we draw Architecture Diagrams corresponding to that DFD, based on Hatley and Pirbhai's method [5].

Architecture Diagrams are an alternate way of looking at and partitioning a system. Just as DFDs, they decompose a system in a hierarchical fashion. Architecture Flow Diagrams represent the assignment of system processes (within a DFD) to locations (modules). Architecture Interconnect Diagrams specify the interconnections between the modules (representing DFD data flows).

We find that requirements are driven by what seems possible and economical to implement. Architecture Diagrams show this more clearly than DFDs. Through reviews, they feed back into the refinement of the system specification.

Additional Requirements

Requirements beyond system functionality have been added to our S/A methodology by attaching notes to a DFD within the CASE tool. All of the relevant notes are incorporated into the specification document when it is automatically assembled.

Three types of requirements are modeled: timing, usability, and reliability. We attempt to quantify these requirements whenever possible: a typical measurement of what today's HEP systems deliver, the quantity we would like to achieve for DSS, and the best measurement we have encountered. When these measurements are available, some appreciation of what requirements are achievable can be gained.

Timing requirements specify critical timings in the system. These are not restricted to just one process, but can show how different aspects of the system must interact within a time constraint. For example, photometric images must be processed to identify galaxies and quasars within two weeks, in time for spectroscopic observations during the next moonless period.

Usability requirements define the ease with which the system can be used and understood. They increase the time that the system is truly available for useful work. An example of a usability requirement is "time to train a new operator to run the photometry data acquisition system."

Reliability requirements specify the availability of the system. For example, the data acquisition system must be available 90% of the time due to the limited number of good nights for observations per year.

Notations

We support adding comments about deficiencies in the model. These comments may be attached to either a DFD or an Architecture Diagram within the CASE tool (just as the *Additional Requirements* discussed above). Automatically including this information in the draft specification allows an overall appreciation of what is missing in the model, provoking responses from the collaborators. Comments are accumulated and researched, allowing us to identify where additional effort is needed most. They may also be printed out separately and distributed for review amongst the collaborators.

Besides comments about deficiencies, textual statements of facts relevant to the specification can also be incorporated into the model. These also are automatically included in the specification document when it is generated, and are available to reviewers of the specification.

Once comments about deficiencies are resolved, they can be excluded from the document. Or, they can be retained in the document and recategorized as textual statements of facts or one of the additional requirements (timing, usability, or reliability).

IV. The Specification Document

Overall, the specification documentation is broken up into three volumes: Data Acquisition System, Data Reduction and Archive System, and Decision System. Each volume has the same overall format:

Overview

- Methods tutorial
- Context level of the system
- First level of decomposition of the system

Requirements

- Depth-first presentation of DFDs and Process Specifications

Architecture

- Depth-first presentation of Architecture Diagrams

Appendices

- Comprehensive Data Dictionary (with cross references)

The Overview section introduces the top level of the system. We have chosen to organise our specification around DFDs, which give a hierarchical view of the system. Top levels are very abstract and are used for overall review. Bottom levels are more concrete and need review by specialists.

The Overview also includes an introductory chapter to help interpret the CASE diagram syntax correctly. Thus, the readers will be able to concentrate on the diagram's content rather than its form. After reading the introductory chapter, readers only needs to look at the chapters describing the portion of the system in which they are interested. In this way, reviewers are not overwhelmed with a large document; they can provide quick responses indicating how well we have understood their particular requirements.

The Overview section has led to a degree of redundancy between the documents. But, because the information about the top levels of the system resides in the CASE tool, only a single copy of the underlying information actually needs to be maintained.

In the Requirements section, our technique is to present the DFDs, one per chapter, with supplemental information. A chapter explains a DFD with the following information:

- text to explain the DFD
- the DFD itself
- comments indicating problems with the DFD
- reliability requirements
- timing requirements
- usability requirements
- other facts appropriate to the DFD
- listing of the Data Dictionary Entries for the DFD
- the Control Specification for the DFD (if any)

In the Architecture section, chapters relate requirements (from the DFDs) to Architecture Diagrams. The Architecture Diagrams are also hierarchical. For each level of the system, there are two architecture drawings: one specifying modules, which can be decomposed further, and another specifying the interconnections between modules.

We present the architecture chapters in the following way:

- Architecture Flow Diagrams
- Architecture Interconnect Diagrams
- comments about the architecture

In the Appendices, each volume of the specification documentation contains a comprehensive Data Dictionary. Each Data Dictionary Entry is supplemented by a cross reference of

- which DFDs it appears in
- which DDEs reference it

V. Interpreting the Model Clearly

Books written as late as 1987 [5] specify methods which are inadequate for our needs. While S/A diagrams are not ideal, their popularity has led to refinements in their use. For example, **team*work*/SIM™** [6] is a commercial package which allows DFDs to drive a dynamic simulation of a system. Because DFDs are used as part of a simulation program, ambiguities in them have been resolved.

As an example, a store is not precisely defined by the CASE tool we use. Different authors have assigned different meanings to the symbol. Hatley and Pirbhai [5] say a store holds a single record which may be re-read until a new record is loaded into the store. DeMarco [3] leaves unspecified how a store is filled and emptied.

We have found that a "Hatley and Pirbhai" store is inadequate, but needed a more rigourous definition than a "DeMarco" store. **Team*work*/SIM™** has provided an acceptable resolution of the problem by binding the nondestructive or destructive read behavior to the data flows emanating from a store, rather than to the store itself.

VI. Tools

Following computing trends at Fermilab, our system is implemented in an X Window System™/UNIX™ environment, with VAX/VMS™ systems used only as X Window servers. Our basic tool for Structured Analysis is **team*work*®** from Cadre Technologies Inc. Architecture

Diagrams are made with the public domain drawing program *xfig*. Document production is done using LaTeX, also in the public domain.

We have developed UNIX™ *awk* scripts to help generate the specification document. For chapters explaining a DFD, we automatically generate files which input:

- comments associated with the DFDs
- timing, reliability, and usability requirements
- other facts appropriate to the DFD
- comments associated with Architecture Diagrams
- Data Dictionary Entries referenced by the DFD

As a result, one rarely has to modify the LaTeX source document. Most information is maintained in the CASE tool.

The specification documents are produced in POSTSCRIPT™ format and may be distributed electronically. They may be viewed online by means of a POSTSCRIPT™ previewer, such as *ghostscript*.[1]

VII. Results

We have met our goals in using CASE tools for DSS:

- compensating for tool shortcomings, such as associating textual notes with drawings
- communicating cleary with the collaborators.

These goals have been met by achieving the following:

- automatic generation of the specification documents
- maintaining most of the information in the CASE tool
- allowing many people to work simultaneously
- supplementing the S/A methods with
    - Architecture Diagrams which provide an alternate view of the system and feed back into the model revisions
    - notes which keep track of deficiencies, ideas, etc., in one central location. The notes change as the model evolves.

These achievements provide a better overall system for catching bugs early in the analysis. This requires additional time now, but will reduce design, implementation, and system debugging times considerably in the future. In the early stages of analysis, we did find up to 25 errors per page of the specification document; after a few reviews, that number had been greatly reduced.

We have found that Structured Analysis' requirement for the hierarchical decomposition of the system is quite useful. Although hierarchical decomposition does extend the time needed for analysis, we benefit from the additional work in that it forms a basis for explaining a complicated system and assists in the organization of the specification. We and the collaborators have found our methods to be successful.

[1] © 1990 by Aladdin Enterprises.

VIII. References

[1] *A Digital Sky Survey of the Northern Galactic Cap* (December 20, 1990).

[2] Vicky White, "Distributed Data Acquisition Systems (PAN-DA) for Fermilab Experiments," *Proceedings of the International Conference on Computing in High Energy Physics / Oxford, 10-14 April 1989* (North-Holland, 1989).

[3] Tom DeMarco, *Structured Analysis and System Specification* (YOURDON Press, 1979).

[4] Eileen Berman, Tom Nicinski, Don Petravick, and Ron Rechenmacher, *Digital Sky Survey System Requirements and Architecture / Volume I / Data Acquisition System*, Draft (Fermilab, June, 1991).

[5] Derek J. Hatley and Imtiaz A. Pirbhai, *Strategies for Real-Time System Specification* (Dorset House, 1987).

[6] Team*work*/SIM™ *User's Guide* (Cadre Technologies Inc., 1990).

[7] Tom Gilb, "Advanced Defect Prevention Using Inspection, Testing & Field Data as a Base," *American Programmer* (Vol. 4, no. 5, May, 1991).