THESIS REPORT
Master's Degree
Institute for Systems Research

An Analysis and Design Methodology for Document Imaging Systems

by L.L. Oaks
Advisor: A.R. Hevner

The Institute for Systems Research is supported by the National Science Foundation Engineering Research Center Program (NSFD CD 8803012), Industry and the University

M.S. 93-5

ABSTRACT

Title of Thesis:	AN ANALYSIS AND DESIGN METHODOLOGY FOR DOCUMENT IMAGING SYSTEMS
Name of Degree Candidate:	Linn Lanette Oaks
Degree and Year:	Master of Science, Systems Engineering, 1993
Thesis directed by:	Professor Alan R. Hevner Co-Director Systems Engineering Program

This thesis proposes that Document Imaging Systems (DISs) provide unique challenges to the system developer. Unlike most information management systems, DISs present features that are not well supported through traditional analysis and design methodologies. A review of several well-known methodologies identifies strengths and weaknesses. DIS features that require special support during system development are presented and an analysis and design methodology is proposed to provide the flexible, yet controlled, environment that is needed.

A case study is used to present the application of the proposed methodology in three areas of specific interest for a new DIS. The importance of early consideration of the unique and complex features of a DIS is emphasized.

AN ANALYSIS AND DESIGN METHODOLOGY

FOR DOCUMENT IMAGING SYSTEMS

by

Linn Lanette Oaks

Thesis submitted to the Faculty of the Graduate School
of The University of Maryland in partial fulfillment
of the requirements for the degree of
Master of Science
1993

Advisory Committee:

Professor Alan R. Hevner, Chairman/Advisor
Professor Mark Austin
Professor Ionnis Minis

ACKNOWLEDGMENTS

I wish to express my special thanks to Richard Neidich, who provided a sounding board for my concepts as I developed my thesis and who provided his guidance in the use of the REDAC software; to Marina Korizis, who was always available to review my writing and to encourage me to keep going when I felt I couldn't make it; to my husband, Stanley C. Oaks, Jr., whose expert editing skills were so necessary and whose patience, support, and understanding never ran out; to my daughter, Deborah, for her assistance with the graphics; and to both of my daughters, Deborah and Kimberly, with whom I now hope to spend more time.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1. INTRODUCTION

For over a decade, prognosticators have been announcing the advent of the "paperless office." Almost as if to mock those prophets, offices around the world are producing more paper than ever before. During 1991 for example, it is estimated that in the United States alone corporations generated "600 million pieces of printout, 234 million photocopies, 76 million letters and 21 million documents each business day" (Cinnamon and Nees 1991).

People create and print countless letters, memos, reports, spreadsheets, and electronic mail messages. Database management systems and decision support systems spit out reams of paper-based reports. Rather than being freed from paper by our computers, we are being buried by the paper they generate. Unfortunately, the huge volume of paper does not ensure that we are getting better information, only more of it (Killen 1991).

Some experts feel that document imaging systems (DISs) provide a vehicle for lightening the burden of paper handling and making it easier to gain efficient use of the information embedded in paper documents. Initially, DISs were viewed as replacements for physical file cabinets. Rather than adding a paper document to a file folder stored in a file cabinet, the document is digitized, indexed to

facilitate retrieval, and stored on optical or magnetic media.

Forward-thinking implementors of document imaging see DISs as systems to help improve customer service by restructuring or reengineering their business processes. Procedures can be streamlined and speeded up by removing the additional overhead of controlling, filing, and retrieving paper documents. All of these overhead activities are necessary to maintain control of the paper documents, but do not contribute to the real work of an organization. United Services Automobile Association (USAA) was one of the pioneers in the use of document imaging (Wallace 1992). Other insurance companies, such as ITT Hartford Insurance Group (Wallace 1992) and Financial Guaranty Insurance Company (Middleton 1992), have installed DISs to reduce paper handling and stay competitive.

Document Imaging

The advent of certain key technologies and the rapid decline in the costs of processing power have helped to justify the cost of implementing DISs. Optical disks, robotics, charge-coupled device (CCD) arrays, high-resolution displays, laser printers, and sophisticated compression algorithms have each played a significant role in making DISs practical for the business environment.

Figure 1 depicts a DIS with commonly configured hardware components.

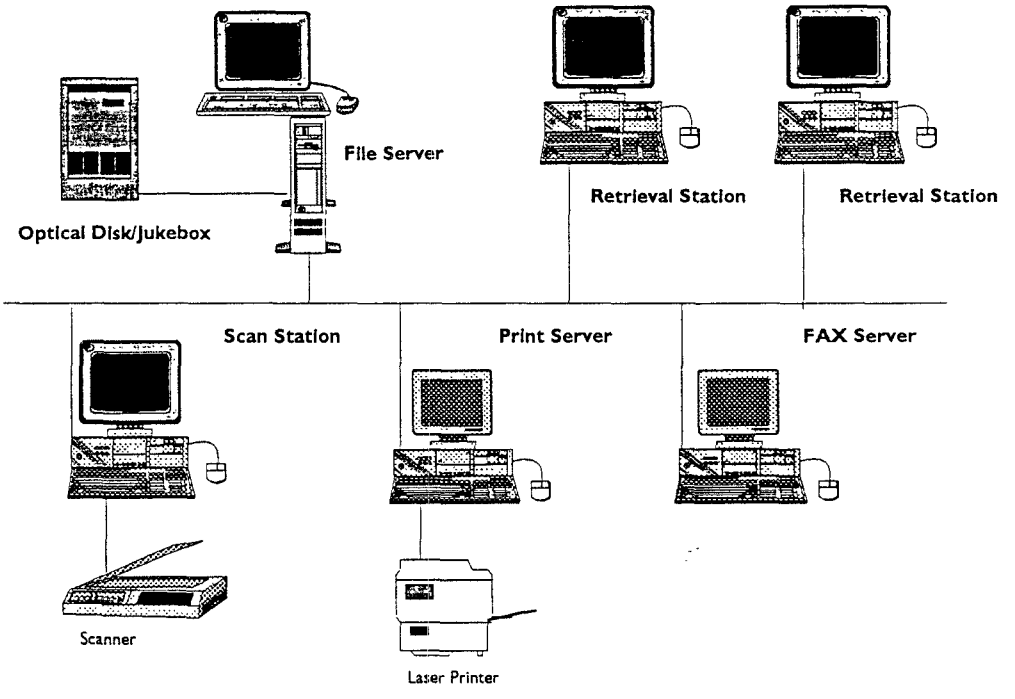


Figure 1. A Typical DIS Configuration

Data flows in traditional information processing systems are centered around four basic activities: capture, processing, storage, and retrieval. While these same activities are present in a DIS, the focus is on document flow and the use of information that is embedded in the document. Documents require special handling, because their value as data is not readily accessible as an electronic image.

Document capture today involves mostly paper scanners and facsimile equipment, although other conversion methods are available. Creating an electronic (raster) image of a document page can result in a one megabyte (1MB) file for a standard 8-1/2" x 11" piece of paper, so compression algorithms must be used to shrink the file size. (A comparison of potential file sizes and compression ratios is presented in Chapter 3.) Because compression is such a compute intensive process, hardware accelerators are frequently used to speed up the process. To support a specific application's performance requirements, image files may be maintained in cache on magnetic disk for hours, days, weeks, or months. However, permanent storage of images most commonly uses some form of optical media (e.g., magneto-optical, rewritable, or write-once-read-many). When tens of thousands or millions of images must be accessible, optical jukeboxes are used to provide near-line storage. Users who must access images frequently benefit from the use of large screen, high-resolution monitors where the displayed image is easy to read, reducing eye-strain. And laser printers produce "copies" of the document that are virtually identical to the original. Images can also be output via facsimile.

Other enabling technologies include optical character recognition (OCR) for machine-printed text and, more recently, intelligent character recognition (ICR) for hand-

printed text. Both OCR and ICR are important for DISs because the words and numbers on a document page are not directly available for querying when the document page is converted to a raster image. Character recognition facilitates automatic indexing that improves the document capture process by reducing the amount of manual keying required to make a document retrievable.

All of this technology is of little use without image management system (IMS) software. Some of the IMS products are very simple, offering a proprietary indexing scheme and basic search and retrieval capabilities. More sophisticated systems use open database management systems (e.g., Oracle, Informix), support complex caching algorithms to improve performance, and provide dynamic workflow processing functions. Workflow capabilities are extremely important for streamlining business processes by providing a mechanism to automatically control document routing and handling.

Integration of all these components requires careful analysis and design that considers the processes and people first, followed by the technology. DISs are not new and for decades all sorts of business information systems have been implemented using proven methodologies. So, why do we need a new or enhanced methodology for analysis and design of DISs? It is our belief that a DIS presents a fundamentally different type of computer-based system which

requires a special approach to achieve successful systems analysis and design. My hypothesis is three-fold.

- DISs have the potential to radically transform the way in which an organization carries out its mission. We have the potential to do more than just automate a manual process. While there are many similarities between developing DISs and other types of business information systems, a DIS can change an operation more dramatically than a new financial system or a new sales tracking system. This thesis proposes a methodology that considers the impact of document imaging on the implementing organization and its business processes.
- A DIS can dramatically change the way employees do their job. Extended periods of time in front of a computer screen are now required. Additionally, a worker accustomed to handling paper documents may have to be trained to interface with a computer, a high-technology skill. The human factors impact of implementing a DIS cannot be overstated, but is frequently not adequately addressed. Most methodologies in use today do not incorporate the evaluation of human factors sufficiently. A DIS methodology must concentrate attention on the individual system users during analysis to ensure that the system is designed to accommodate users' needs,

rather than requiring users to adapt business processes to a rigid system implementation.

- Full use of the capabilities of a DIS frequently cannot be understood until it is placed in the hands of the users. DISs have been available since the late 1980s and the technology is considered to be mature. Many organizations feel it is possible to implement full-blown systems from the outset and may not count on the changes in organization that frequently come with implementing a DIS. A DIS methodology should be based on phased development and prototypes, which are critical to successful implementation of a system that impacts both the organization and the user so dramatically.

Summary

The courses in the Systems Engineering Master of Science curriculum cover topics which are directly applicable to the practice of information systems engineering in government and industry. Many of those topics are incorporated in the contents of this research, including systems engineering principles, information systems analysis and design, and human factors in systems engineering. The primary focus is on the analysis phase of the system development life cycle, because it is unlikely

that a quality system can be built without an accurate understanding of the problem it is supposed to address.

Much of the material found during the literature search for this thesis discusses software development, as opposed to systems development. Systems development encompasses more than just developing software for computer-based systems, however, there are a large number of parallel activities that apply to both. The material covering software development is considered by this author to be highly relevant to systems development, as well. In practice, software development is often a sub-function within the development of a system, as it is when developing a DIS.

Chapter 2 presents a review of system development paradigms and methodologies. It is not all inclusive, but covers many of the well known and a few of the not so well known methodologies.

Since document imaging is still not widely implemented, additional information on the technology and some of the known implications for implementation are included in Chapter 3. This provides a basis for defining the phases of the DIS methodology, as well as the activities to be carried out during each phase.

The experience of professionals who have implemented DISs is extremely valuable in understanding the benefits and drawbacks of particular methodologies and methods. As

part of the research for this project, computer professionals with specific experience in the DIS arena are interviewed. The summation of the interviews and the experiences of these engineers is offered in Chapter 4.

Chapter 5 presents a methodology for use in the development of DISs. The primary focus is on the analysis and design phases of the DIS implementation, but the other phases of the software development life cycle are also considered.

Investigation of three system development activities for a real world system is conducted using the DIS methodology presented in Chapter 5. The tasks undertaken to conduct this analysis are reviewed in Chapter 6. Testing the methodology in this way provides a valuable means for evaluating its applicability and robustness of the methodology for accomplishing stated goals.

Chapter 7, Conclusions and Recommendations, brings together the major points from each of the preceding chapters. Benefits and drawbacks of the proposed DIS methodology and methods are reviewed. Finally, recommendations are presented for further research that could be conducted, with the intention that it will build upon the knowledge gained in this project.

CHAPTER 2. REVIEW OF METHODOLOGIES

Is it necessary to have a methodology for system development? Those in favor of hacking, or ad hoc, system development would probably say no. However, many authors, including Boehm (1988), Cutts (1991), and Yourdon (1992), agree that the system development effort benefits from the use of a defined process for getting from the feasibility study, through the intermediate phases of analysis and design, and on to system development, implementation, and maintenance. This thesis extends the Yourdon (1992) concept that a methodology is akin to a cookbook for developing systems. The methodology lays out the ingredients (methods) and the system development process model provides the steps (phases) that guide the developer to the goals. Like most recipes, both the process and the methodology can be adjusted to taste and still provide a successful implementation.

A discussion of system development process paradigms is presented next, followed by a review of methodologies.

System Development Process Models

The process model provides the ordering of activities that are to be conducted during system development. According to Boehm (1988), there are two topics the process

model must address: selecting the next activity and determining how long to conduct that activity (a schedule).

The predominant paradigms are the waterfall and spiral models. The code-and-fix model will not be discussed. Even though it is frequently used for systems/software development, it does not provide any guidance or structure for the developer.

Waterfall

The Waterfall system development process model, or a variation of it, is probably the most frequently used by developers. It grew out of the structured programming movement, which evolved to incorporate structured analysis and design. While different authors include varying numbers of phases, the model generally follows the flow presented in Figure 2 (DeGrace and Stahl 1990).

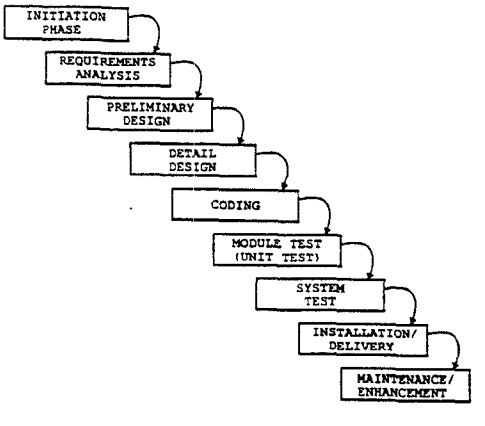


Figure 2. Traditional Waterfall Model

Benefits of the Waterfall approach to system development are primarily seen or perceived by an organization's management. By prescribing the Waterfall model, management generally feels that they have imposed some order and control over the process (DeGrace and Stahl 1990). The status of the process can be monitored by discrete events that mark the beginning and end of each phase. These events generally involve formal documents that are produced, reviewed, and signed as an explicit acknowledgment of the end of one phase and the authorization to proceed to the next.

However, the formality of the process and the rigorous documentation requirements are also a drawback to using the Waterfall model. It originated at a time when the price of the computer was far in excess of the salaries of the development staff (DeGrace and Stahl 1990). With the current cost of hiring and maintaining a qualified staff, the cost of producing the documentation alone can exceed the cost of the computer on which the system will be implemented. It is also possible to explicitly follow the Waterfall model to produce all of the required deliverables, go through all of the reviews, gain all of the necessary approvals, and find out in the end that the wrong system was defined (Boehm 1988). Poorly stated requirements or poor communication between the developers and users are common reasons for this type of failure.

Another problem with the Waterfall model is the fact that there is nothing, with the exception of paper diagrams and descriptions, to show to the users until fairly far into the project. By the time the users begin seeing the screens and system functions that are being developed, the cost of making changes is incrementally higher than it would be if the user were able to provide input at an earlier stage.

The Waterfall paradigm is considered too inflexible for dynamic systems, where change is the rule rather than the exception. If followed precisely, the process of developing a system using this approach can take so long that the system is never completed due to the evolution of the system requirements.

Organizations commonly modify the Waterfall model to make it more palatable and reduce the amount of time it takes to complete the development process. Boehm (1988) refers to the addition of verification or feedback loops and prototyping as one mechanism for making it more responsive and flexible. This helps resolve issues related to late or evolving requirements definitions, but still leaves the burden of excessive documentation. To meet their specific needs, some organizations remove activities or simplify deliverables.

Spiral

While the Waterfall process model results in a document-driven or code-driven approach to system development, the Boehm spiral model (Boehm 1988), as seen in Figure 3, is risk-driven. The life cycle involves repeated loops that spiral outward from the center with each loop being composed of a series of activities. The goals for each loop are to:

- determine the objectives for that cycle
- consider alternatives for achieving those objectives
- select the alternative that best achieves the goals and reduces the identified risks
- determine the next steps based on the results of that cycle

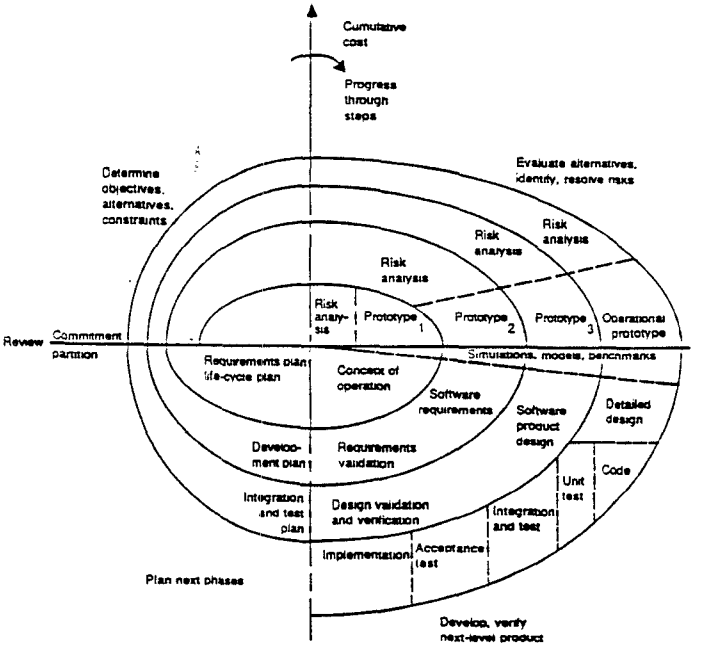


Figure 3. Boehm Spiral Model

Boehm (1988) cites one of the benefits of the spiral model to be its ability to accommodate a variety of system development methodologies, as appropriate, within its cycles. Thus the Waterfall model can be used to provide detailed documentation for users who feel more comfortable with more paper, or prototyping can be used where it is deemed critical that user reactions be measured before the design proceeds. The end result is that the activities of each cycle are customized to address the specific objectives established at the beginning of that cycle, based on the level of risk assigned to those objectives.

However, the spiral model is not without its problems. Since the government and much of private industry is so well indoctrinated into the Waterfall and structured methodologies for system development, there is not normally the flexibility to approach the project with a risk-driven focus used by the spiral model (Boehm 1988). The customer is looking for fixed deliverables to measure progress and, while that may map into a specific cycle of the spiral, it does not fit well for the entire project approach.

A second problem relates to the difficulty of accurately assessing risk. Identifying the level of risk of failure for a given system function and deciding that the risk is minimized by performing a particular activity, such as producing a prototype, carries an element of risk with it as well. The result of under-estimating the risk

can be a product that superficially meets specifications, but does not satisfy requirements that might be uncovered by performing a detailed analysis and design.

A variation of the spiral, presented by Mills et al, (1986), can be described as a series of connected loops (see Figure 4), similar to a Slinky® toy. In this enhancement of the spiral model, each loop is composed of three phases:

- planning -- identifying the issues, considering alternatives, and setting the course and the goals for this loop
- execution -- carrying out the activities identified during planning
- evaluation -- reviewing the results of executing the activities relative to the goals established during planning

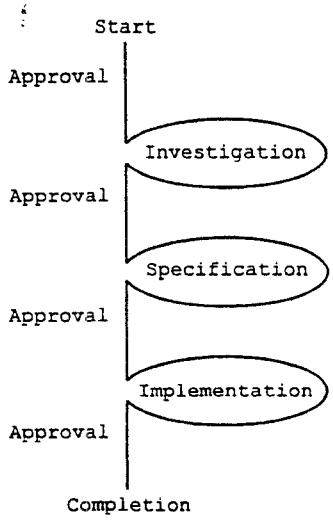


Figure 4. Mills Spiral Model

The loops are iterations of three classes of activity: investigation, specification, and implementation (Mills, Linger, and Hevner 1987). The order and number of repetitions of these loops vary according to the requirements of the project. There can also be parallel spirals that occur concurrently, each separate spiral having its own unique sequence of activity classes.

The Mills spiral provides a great deal more flexibility than the Boehm model. While activities could be dropped from or added to each cycle of the Boehm spiral, it does not easily accommodate recursive execution of activities. Since the Mills spiral accommodates activities in any sequence, recursion and repetitive loops are supported, as needed.

As with the Boehm spiral, the flexibility of the Mills spiral development process is not easily accepted in government contracting environments or in many private companies. The expectation, or the mandate, that process activities and milestones be laid out in detail up front, with penalties for not meeting those goals, deters many contractors from attempting to use these spiral paradigms. Additionally, the Waterfall model requires that detailed documentation be generated for each component, leaving no flexibility for producing risk-leveled documentation. Consequently, the same level of detail is used to define both very complex features and standard, well-understood

features, even though the possibility of misunderstanding the complex feature is far greater.

The flexibility of the spiral models allows system developers to address complex systems incrementally (Hevner 1992). Attacking highly complex problems without breaking them down can lead to failure of the system or even failure to get the system implemented. The use of the incremental approach actually provides control over a project that is too large to attack in its entirety with the Waterfall process model.

Both spiral models involve user and developer reviews of the activities to gain consensus. At the end of each loop, the reviews validate work that has been accomplished and establish support from all participants for the process to continue.

A lack of understanding of the spiral paradigm by users and system developers can make it difficult to implement. The authors of these models are still advancing their designs, providing more detail on how to use them. To date, there is not yet a large constituency for the use of spirals for system development and there are few automated tools that support them.

Defining Methodology

One fact becomes abundantly clear when researching the topic of system development: there are almost as many

definitions of terms, and as many different approaches, as there are authors on the subject. In proposing a methodology for developing DISs, it is important to define the terms methodology and method. Some authors would argue that methodology is used to make a process sound complex, something requiring an expert to interpret and worth "attending a seminar [put] on by a major software guru" (Constantine 1989). To quote Webster's New World Dictionary (1986), methodology is:

> "1. the science of method, or orderly arrangement; specifically, the branch of logic concerned with the application of the principles of reasoning to scientific and philosophical inquiry 2. a system of methods, as in any particular science"

In the context of this thesis, a methodology is defined as a set of methods that are organized to guide the system developer through the various stages of development.

A method, then, is a way of doing something within the context of a methodology. The methods appropriate for use in the various phases of system development may include diagramming techniques, but a diagramming technique cannot be considered to be a complete methodology.

Three aspects of a system which are generally considered, with varying levels of emphasis, in a

methodology are: process, data, and control. These three aspects are fairly well understood and agreed upon by methodology gurus. Process, or function, is what the system is doing. Data is what is being processed. And control, or timing, is "when" the processing is to be done.

In an article by K. W. Short (1991), an additional three aspects are attributed to Zachman (1987): rules, location, and people. Rules are important because there may be institutional requirements, or constraints, that dictate why something is done a certain way. Where something is done or where the data is stored and processed (e.g., location) must also be considered when analyzing and designing a system. This is becoming more important as networks and distributed system architectures, such as the client-server model, are implemented. And, as will be elaborated in later chapters of this thesis, understanding the people and their role within the system, is key to the success or failure of any system being implemented, but is especially important with imaging system implementations.

Methodologies

This is not a comprehensive review of analysis and design methodologies as it would be impractical to include all published methodologies in this thesis. A cross-

section of those encountered while researching this topic are described. For each one described below, the benefits and drawbacks of using that methodology are presented, hopefully with an unbiased viewpoint.

Process-oriented methodologies

Early proponents of the structured movement included Yourdon, Constantine, and DeMarco. The Yourdon Structured Methodology™ (YSM), in its first iteration, was process-oriented and emphasized the use of structure-charts and data flow graphs to depict the system design. Structured design concepts published by Yourdon and Constantine (1978) formed the basis for YSM. Tom DeMarco (1978) wrote about applying structured techniques to the analysis phase. Data flow diagrams, decision tables, data dictionaries, and structured English were presented as tools to help the systems analyst prepare a system specification. YSM has evolved over the years into an "integrated method" (Bowles 1990). The most recent version of YSM now recognizes the importance of data modeling. Recently, YSM has been enhanced to include object-oriented systems design.

Other methodologies that use the same process-oriented approach include Structured Systems Analysis and Design Methodology (SSADM) (Cutts 1991) and Structured Analysis and Design Technique™ (SADT) (Marca and McGowan 1988). Jackson Structured Design (Henderson-Sellers and Edwards

1990), is also considered to be in this class of methodology.

Data-oriented methodologies

A data-oriented approach to analysis and design is based on the concept that data is more stable than the processes for any given organization. It also assumes that data can and should be analyzed independent of the current processes that use the data.

Information Engineering (IE) is a data-oriented methodology which uses five classification criteria to analyze a system, in both its present and future forms. Once each classification is defined and clearly documented, the results can be evaluated on an organization-wide basis to determine strategies for implementing the new system. Information Engineering is designed to address an organization or enterprise in its entirety, whereas most other methodologies address an application or system.

The five classifications (Short 1991) of IE are: business functions, management categories, technology types, technology usage within groups, and problem complexity. IE looks at business strategies, as well as data and processes. Using the IE methodology, the system developer attempts to determine whether the data and processes are required for strategic, planning, controlling, or operational reasons. These factors all

affect the type of system that should be implemented. Short further divides technology types into industrial automation (e.g., process control), messaging (e.g., electronic mail), professional automation (e.g., spreadsheets and word processing), and transaction processing (e.g., databases). The specific technology is identified as being either common services, shared at a local level, or available to a single user or piece of equipment. Finally, the complexity of each process or problem is examined. Low complexity problems with well-defined, well-structured policies and procedures are placed at one end of the spectrum. Unstructured and ambiguous problems with no clear solution are grouped at the other end and present a greater challenge to the system developer.

IE does not fully address the issue of control, largely because it has been felt that "businesses are not controlled, but are a collection of asynchronous processes triggered by human or informal events" (Short 1991). Even though an attempt has been made to encompass the six aspects of system definition discussed in the overview to this chapter, the IE methodology is still not fully developed in the areas of location and people.

The James Martin IE methodology divides a system development project into four phases: strategy, analysis, system design, and construction (see Figure 5). Both data

and process (activities) are included in the phases, but the emphasis is on understanding the data requirements.

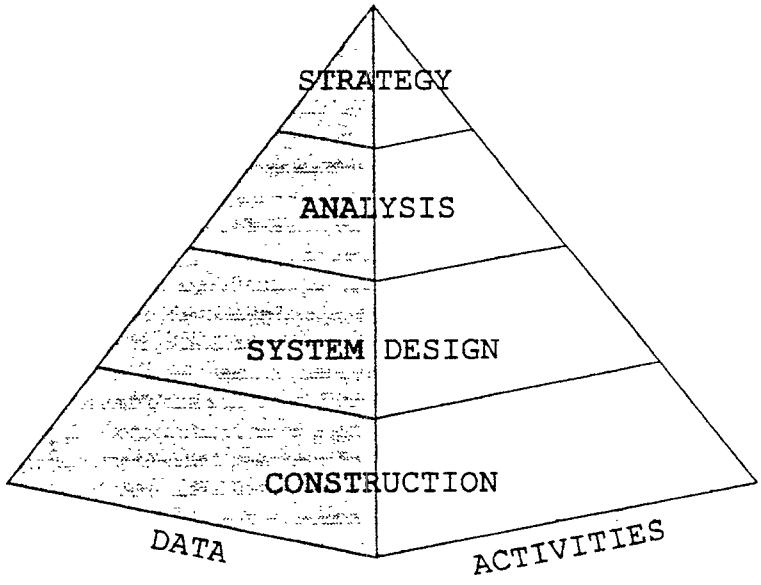


Figure 5. James Martin's Pyramid

Object-Oriented

Coad and Yourdon (1991a) present object-oriented analysis (OOA) as a "merging of disciplines." They contend that OOA encompasses the strongest concepts of the other methodologies and, therefore, provides the best approach to dealing with the complexity of system development today. Other than entity-relationship diagrams and semantic data modeling, however, they draw very little from the traditional methodologies, as can be seen from the diagram

(see Figure 6), which was presented in the Coad and Yourdon book.

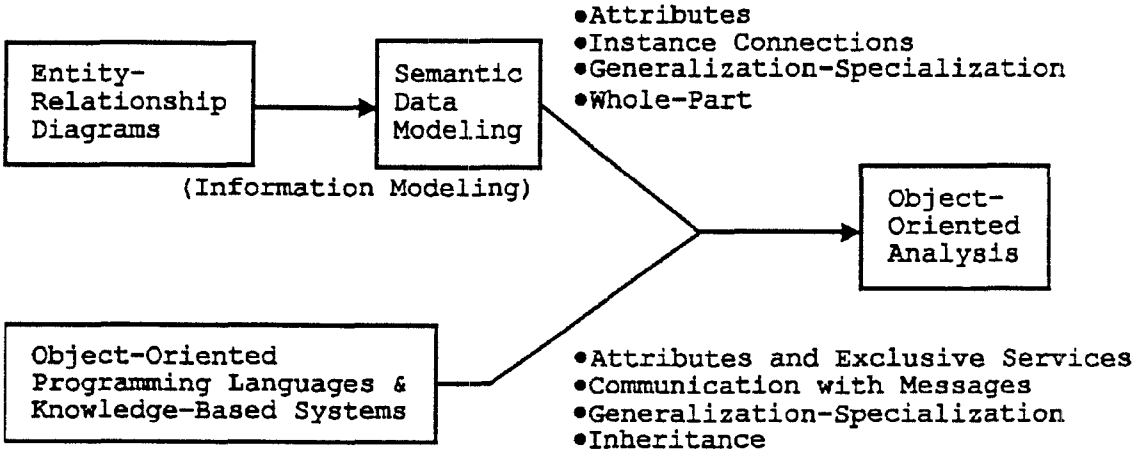


Figure 6. Coad and Yourdon "Merging of disciplines"

A separate text by Yourdon and Coad addresses the object-oriented design (OOD) issues as they relate to a system development methodology (Coad and Yourdon 1991b).

Within the object-oriented paradigm, a system is a set of objects which interact with each other, providing services under the client-server model (Henderson-Sellers and Edwards 1990). An OOA/OOD methodology introduces new terminology and a new way of viewing systems and the system development process. A few of the new terms include: object, encapsulation, class, inheritance, and polymorphism. The definitions of these OOA/OOD terms are beyond the scope of this thesis.

The object-oriented (OO) methodology provides the system developer with one model that is used throughout the analysis and design phases of the life cycle, unlike the more traditional methodologies that use different techniques and models for each of the different phases (Korson and McGregor 1990). This consistency is viewed as a benefit when transitioning between phases. The risk of making errors during the transition between analysis and design is minimized since the OOA/OOD techniques are consistent across phases. When using structured methodologies, such errors are easily made during the translation from the process or data model in the analysis phase to the structure or hierarchical model in the design phase. Some of the structured diagramming techniques, such as data flow diagrams, can be used during OOA/OOD to define what happens within an object or class, but are never used on a system level. Some of the models used in OOD are: object-relationship graphs, client-server diagrams, inheritance charts, or collaboration graphs (Henderson-Sellers and Edwards 1990).

Software reuse is supported by the OO methodology to a greater extent than by any of the traditional methodologies. In fact, one of the tasks specified for a system developer is to identify those objects in the design for which existing objects can be incorporated either directly or by extending them.

Henderson-Sellers and Edwards (1990) propose a seven point methodological framework for OO system development. They start with a requirements specification, which may include "timing details, hardware usage, cost estimates, and other documentation." For each requirement identified in the specification, they suggest that objects and services be defined. Next the developer determines the interactions that should take place between objects. At this point, the process transitions from analysis to design. Now the developer addresses establishing the appropriate object classes and identifies objects available for reuse. Hierarchical relationships between classes are defined and, finally, the classes are generalized, to promote further reuse.

All of this leads, according to OO proponents, to a methodology that provides a great deal of flexibility. OOA/OOD does not pin down the structure of entities in the early stages of the design, thus allowing it to be molded by the discoveries made as the project progresses. Another claim of the OO group is that OO methodology allows programmers to begin their work earlier in the process, without fear that the entire implementation will have to be thrown out or massively changed if known requirements change or new requirements are uncovered. The manner in which the objects and classes are defined leads to modules

that can remain useful with minimal changes or can be discarded/replaced without impacting total design.

One of the major drawbacks to the OO approach is the very steep learning curve for systems developers. Since the entire approach to viewing and developing systems differs so radically from the structured approach, there are not many people who are skilled at the OO methodology. This problem will lessen over time, if the methodology catches on and new engineers and programmers are trained to use OOA and OOD. Although there is a growing belief that OO is better suited to "complex new forms of integrated systems" (Fichman and Kemerer 1992), system developers must first become skilled in the process of identifying and designing the components of the system -- objects.

Other Methodologies

As mentioned earlier, while other methodologies exist, it is not feasible to describe each one within the context of this research report. In addition to those already described, others deserve mention, and may be found useful within the context of the proposed methodology for DISs. These additional methodologies include: cleanroom, prototyping, and all-at-once.

- Cleanroom -- The basic premise of the cleanroom methodology is that an ounce of prevention is worth a pound of cure. It would be unthinkable for a

structural engineer to allow a bridge to be built with a design flaw, but software engineers knowingly create software without rigorous verification of correctness. The goal should be to get it (e.g., system development) right the first time. The cleanroom methodology is very formal and stresses the importance of using principles that have been used successfully in other engineering disciplines to develop "error-free" designs. The diagram of the cleanroom system development model presented by Hevner, et al. (1992), depicts the activities and teams that participate in the process (see Figure 7).

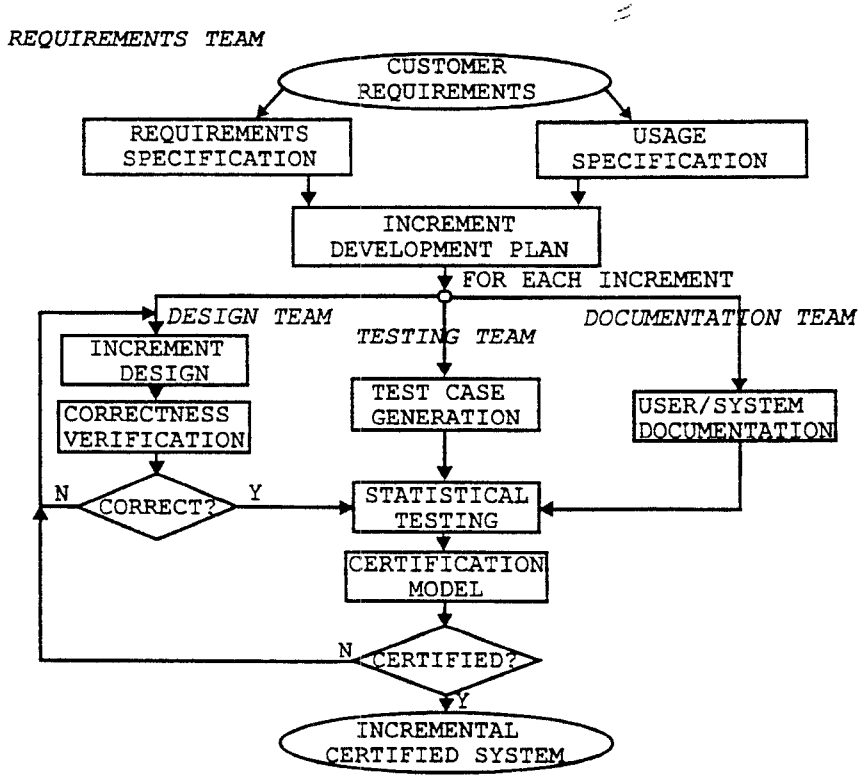


Figure 7. Cleanroom Development Process

The teams are used to implement the cleanroom for systems analysis and design, with separate tasks assigned to each: specification, development, certification (Cobb and Mills 1990), and documentation (Hevner, Vagoun, and Lemmon 1992 and Hevner 1992). The first team develops the system specification document, which describes the requirements of the system. This document is used by the development team to design the system, the certification team to design the test suites, and the documentation team to develop the system documentation. An incremental approach is used to retain control over development of the system. Fully functional subsystems, which provide the required functionality, are specified, developed, and documented. Testing is a parallel activity used to certify the subsystems function as specified.

Box structures are used within the cleanroom methodology to define the system. The box description graphic (BDG) and the box definition language (BDL) provide a diagramming technique and a formal language for documenting the analysis and design. The black box, state box, and clear box are used for system decomposition (Mills, Linger, and Hevner 1987) and provide a formal, mathematics-based technique for system development. Thus, the cleanroom methodology

maintains its rigorous, verification-based approach to system development.

Quality is the essence of the cleanroom methodology. Every team is indoctrinated with the concept that quality comes first. Other methodologies include reviews at various stages of the design, but with cleanroom the design reviews and code walkthroughs take on a new seriousness. The testing is not done to find "bugs," but to statistically determine the mean-time-to-failure (MTTF) for the system (DeGrace and Stahl 1990).

While quality is seen as a primary benefit of using the cleanroom methodology, the inherent control and incremental approach to system development provide the additional benefit of helping to control costs.

Significant gains in quality are possible by implementing cleanroom methodology, but some authors worry about its "military nature" (DeGrace and Stahl 1990). There is concern that the creative element to system development and the artistic nature of programmers will be quashed by the rigidness of the cleanroom approach. Until the rigors of the methodology and importance of developing "error-free" systems are taught within the systems engineering discipline, just as it is for other engineering

disciplines, it seems unlikely that the cleanroom methodology will gain a very large audience.

- Prototyping -- While prototyping is occasionally promoted as a full life cycle methodology, it is most useful as a technique within a methodology. A small, discrete system can be designed and implemented exclusively with prototyping. But few systems today remain small or isolated over time. Documentation for prototyping, where it exists, typically reflects the system as developed and is not usually at the same level of detail as the evolutionary documentation found in other analysis and design methodologies (DeGrace and Stahl 1990). The formal definition of the prototyping methodology includes a feasibility study, along with a study of the existing system. However, the rest of the model revolves around defining, building, and testing the prototype. Once the prototype is determined to exhibit the features and functions required for the new system, the prototype is converted to a production system and installed.
- All-at-once -- DeGrace and Stahl (1990) describe two all-at-once team approaches to system development, where many tasks from the traditional methodologies are done concurrently, or in parallel.

One approach, the sashimi team, uses a compacted version of the Waterfall model with increased communication between the adjoining phases.

The second team approach is the scrum version of all-at-once development, commonly used in Japan, which includes a representative from a variety of groups: designers, coders, testers, and users. They are brought together and told that they have been selected to work as a team to develop a system. Emphasis is placed on their special abilities and how important the project is for the success of the organization. The idea is to promote the concept of the team being able to accomplish something that could not be accomplished individually, i.e., the whole is greater than the sum of its parts. The team is not told how to get to their goal, but is challenged to provide the best solution they can find. Each member contributes to the system development from their own perspective, but soon begins learning how their piece of the project integrates with the other pieces. The authors report that the successes of using this approach have largely exceeded even the original goals of the organizations that have implemented it. They also report that this approach is most suitable for new development and not for re-engineering of an existing system.

The two-man version of the scrum pairs a user with a developer to arrive at a solution to a specific project objective. The user contributes the system knowledge and the developer brings the information system and programming knowledge. Successful use of this technique requires careful selection of the team, since any incompatibility could be a major problem. If either team member becomes intimidated by the other, the system will be reflective of that influence and will likely suffer from either a lack of responsiveness to actual requirements or a design that is overly influenced by the current system. When implemented well, however, it provides an excellent tool for cross-training users to be more understanding of the constraints of system development and for developers to be more understanding of the user's environment and requirements.

A "one person team" approach to all-at-once development may be successful for small projects, but is reminiscent of the early days when systems were hacked together by one developer. In the rare instance where the developer has in-depth understanding of the requirements, as well as skill as a programmer, this technique can be successfully used. In the end, there is likely to be little documentation and the quality of the system will be totally

dependent on a single individual. (DeGrace and Stahl 1990).

Summary

There are many sound principles developed within the available methodologies. Successful and unsuccessful uses of each methodology can be cited. The challenge of this research effort, however, is to identify those specific principles that will best benefit the development of a DIS and provide the greatest level of guidance to the system developers without overburdening them with excessive detail or documentation. The ultimate goal should always be a system that meets users' needs.

CHAPTER 3. DOCUMENT IMAGING SYSTEMS

Over the years in which computers have been widely available to private industry and government for office automation and data processing, information systems development has largely concentrated on "adding automation to a process, rather than automating the process." This has been discussed by a number of authors, including Avedon (1992) and Stadler and Elliott (1992). Now, with the maturing of imaging technology, the opportunity exists to automate the process.

Initially, organizations installed DISs for storage and retrieval of documents -- electronic filing cabinets. A DIS saves floor space and reduces financial expenditures for file cabinets. It speeds up the retrieval of documents, since there is no need for a person to get up from their desk, go to a file room, extract a file folder (providing it can be found in the expected location), and return to their desk. And there is no refiling process. The electronic documents are always available. Since the document's image is not physically removed from the electronic file when a user is viewing it, many users can access it simultaneously (Avedon 1992). This simplistic viewpoint, however, overlooks the possibilities of restructuring business processes to take advantage of the full range of DIS capabilities (Safdie and Flanagan 1992).

Document Image Processing

In the typical DIS, specialized equipment and software is integrated with standard computer components. Components of a DIS that are also configured for office automation and data processing systems include:

- hardware -- central processors, workstations, communications links, printers, magnetic storage
- software -- operating systems, communications, databases

Additionally, DIS-specific hardware and software components, such as scanners and optical drives must be evaluated, selected, and integrated when introducing document imaging. These components cannot realistically be selected until the processes are analyzed and details of capacity and performance are evaluated. The primary processes in a DIS are document capture, storage, and retrieval. Document capture is further broken down into scanning, indexing, quality assurance, and committal.

Scanning

Document capture processes often determine the success or failure of a DIS. Paper documents are converted to electronic form, most commonly raster image files, using scanners. Scanners are available with a variety of features, depending on the documents to be scanned. There

are a number of questions that are relevant to the selection of scanners. Those listed below are extracted from the list of questions prepared by AIIM (1992) and represent only a sampling of the information that must be known. (A more complete list is found in the AIIM RFP Guidelines Technical Report.)

- What sizes and thicknesses of paper are to be scanned?
- Are the documents single or double sided?
- Are the images bi-tonal, grayscale, or color?
- What is the quality of the images?
- What is the condition of the paper?
- How many documents must be scanned within what period of time?

Once the pages of the document are converted, they can be stored on magnetic media. However, the raster image file that is created by scanning a piece of paper is quite large, so it is compressed before storing. The compression ratio between uncompressed and compressed image files is dependent on both the algorithm used and the contents of the scanned page.

Compression algorithms used for DISs are usually CCITT Group III or IV, which were developed for facsimile transmission across communications links. Other, more efficient algorithms are available, but may be proprietary rather than based on a de jure standard. An uncompressed

image file of close to 1MB (8.5" x 11", good quality, typed text) can be expected to compress to less than 50 kilobytes (KB) using CCITT Group IV.

As users begin to store more complex images, such as grayscale and color photographs, in their image databases, the CCITT algorithms do not significantly reduce the size of the file. In some cases, it is even possible that the compressed file size will be greater than the uncompressed file. The JPEG (Joint Photographic Experts Group) standard offers a mechanism to achieve a better compression ratio. However, there is a cost to using JPEG. It is more complex than CCITT and usually requires hardware assists in the form of compression boards with RISC (reduced instruction set computer) processors to provide acceptable performance. The developer must also determine whether the most efficient form of JPEG (lossy), which actually discards bits of the image, can be used to achieve the greatest reduction in file size, or if, perhaps for legal reasons, the lossless or exact version of the algorithm must be used.

Figure 8 provides representative ratios for CCITT and JPEG compression algorithms (Latimer 1992) for 3" x 6" documents (check-sized). In addition to the effect of the chosen algorithm, the table presents a clear picture of the impact of scanning resolution. An increase from 200 dpi to 300 dpi more than doubles the size of the image file before

compression. When a higher resolution is required for legibility of the displayed image, the selection of an appropriate compression algorithm becomes increasingly more important.

Image Type	Resolution (dpi)	Raw Image Size	Compression		Compressed Image Size
			Type	Ratio	
Bi-tonal	200	90KB	CCITT G3 1D	8:01	11.25KB
Bi-tonal	200	90KB	CCITT G3 2D	10:01	9KB
Bi-tonal	200	90KB	CCITT G4 1D	15:01	6KB
Bi-tonal	300	202.5KB	CCITT G4	15:01	13.5KB
8-bit grayscale	200	720KB	JPEG-exact	2:01	360KB
8-bit grayscale	200	720KB	JPEG-lossy	20:01	36KB
8-bit grayscale	300	1620KB	JPEG-lossy	21:01	81KB
4-bit grayscale	100	90KB	JPEG-exact	2:01	45KB
4-bit grayscale	100	90KB	JPEG-lossy	20:01	4.5KB

Figure 8. Typical Compression Ratios

Finally, many file formats are available for storing images, however, Tagged Image File Format (TIFF) is probably the most commonly used. The definition for TIFF was developed by Adobe Systems and release 5.0 of this de facto standard is currently being used by many image management system software packages.

Indexing

The next step in the capture process is indexing. If there is to be any hope of retrieving the document, a database index and/or a full text index of the information in the document must be created. The document index must

contain enough information to allow the document to be identified for retrieval, but not so much that the data entry work becomes overwhelming.

In some cases, the indexing can be done automatically by integrating OCR or ICR technology. This is possible when the document format is consistent, allowing the software to be programmed to know the location on the document page to look for the data. Bar code technology provides another mechanism for partially indexing documents without manual keying. When document types are unknown or varied, the data may have to be keyed into the database record manually. Hybrid systems allow a combination of automatic and manual indexing. A DIS developer must ensure that the indexing does not become a bottleneck in the document capture process.

Quality Assurance

Many organizations include a quality assurance (QA) step in their document capture process. The quality of the images is evaluated and the accuracy of the indexing is checked. Installations that use high speed scanners frequently do not display every image during the scanning process. The QA step then becomes important for assuring the integrity of the image database. Document images that do not meet quality criteria are rescanned. Incorrect index fields are rekeyed.

Committal

Once the quality of the document image and the index is verified, images are ready to be committed to optical media. While image files could be written to optical as soon as they are scanned, there are reasons why that should be deferred.

Many organizations use write-once, read-many (WORM) optical media for image storage. Once an image is written to a WORM optical disk, it is permanent. For performance reasons, it is desirable for the images in a document to be written contiguously to the same optical platter. Since the document image quality is not checked until the QA step, it is possible that one or more pages of the document may have to be rescanned. In a multi-user system, if the images are written to optical when scanned, the rescanned pages would be written to another location on the optical platter, or possibly even to another platter. By deferring the committal process, documents pages are written together on the optical disk, facilitating the retrieval process. This also reduces the permanent loss of optical disk space, which with WORM media cannot be reclaimed if a page has to be rescanned.

Retrieval

For many organizations, the ability to retrieve and display or print document images provides the true value of

the system. In a storage and retrieval system, the user enters a query to select relevant database records. The user then determines which record(s) relate to the document being sought and can request that the image be displayed. There may also be situations where many documents are pertinent and the user will need to see more than one image to complete a task. Important factors to be considered by the DIS developer include requirements for speed of image display, image quality, and format for the user interface.

The document capture and retrieval processes may be affected if a workflow application is implemented. Workflow software may be designed to route the document and related information through a number of processes involving both human interaction and machine processing. The analysis and design of workflow systems tend to be much more involved and complicated, especially where the organization is also trying to implement business process reengineering.

Building A DIS

Many vendors promote document images as merely another data type, similar to character and integer fields in a database. This represents an oversimplification which may benefit the marketing and sales of DISs, but sets unrealistic customer expectations relative to the complexities of implementing a DIS. There are many factors

which are unique to imaging and others that need to be addressed more carefully. They all add considerably to the work of the system developer. The DIS analysis and design methodology must provide appropriate mechanisms which allow these details to be taken into consideration. Several of the DIS factors are described below.

Document Analysis

One of the first activities required during the analysis phase is identification and study of the documents that are to be stored in the DIS. The results will help determine the practical options for design and configuration issues that will arise. Document types and document usage will impact the options for compression algorithms, scanners, and image monitors and controllers. The quantity and content of documents are used in calculating capacity and throughput requirements. Legal requirements for document integrity affect the optical media, optical drives, and compression algorithms that can be used.

While analyzing the physical documents, the developer must determine the definition of what constitutes a document. The answer will vary from system to system and possibly even from person to person within an organization. When dealing with paper, it is easy for a human being to identify a document, but once converted to electronic

format, it is not obvious. In a DIS, a scanned page is stored in a file. A single page can be a document. Or multiple pages can be linked, via the database used by the image management software, to form a document. Document analysis includes defining exactly what a document is and what the document looks like in electronic form.

Image Capacity Analysis

Determining the optical and magnetic capacity for a DIS requires knowledge of several variables:

- the physical document, from Document Analysis
- the volumes of documents to be captured and stored in the DIS
- the type of image compression algorithms supported by the image management system

The size of an uncompressed image file is largely dependent on the contents of the document page. Plain text documents require less storage space than photographs. Black and white requires less than grayscale or color. Obviously, the more documents the system must store, the greater the required disk capacity. And the more effective compression algorithms will yield the most images per megabyte of storage capacity available. At a minimum, image capacity analysis requires examination of these three factors.

User Interface Design

While this is an important consideration in traditional systems, it is critical in a DIS. Users who are accustomed to handling paper files, perhaps in conjunction with an automated system, must now adapt to viewing the paper image on a workstation monitor. Allowing a user to maintain the contextual feel of a paper-based system requires careful analysis of user needs and extensive prototype testing to validate design choices. The display resolution must be sufficient to allow the user to read the information on the image page without eyestrain.

Performance

As with the user interface, performance is a consideration during the analysis and design of all systems. A DIS introduces complicating factors that are not normally present in a data processing system. Optical media is slower than magnetic media. Additionally, with optical jukeboxes, the media may not even be in a drive where it can be read immediately. The system software may have to locate a platter, dismount a platter in a drive, mount the newly requested platter, and finally read the image(s) from the disk. The processes of compressing and decompressing images introduce additional overhead. Compression of images prior to storage impacts performance

during the scanning operation. Decompression impacts any type of retrieval (e.g., display and printing).

In cases where rapid performance is critical, there may not be time to accommodate all of this processing and still achieve required rates. When DISs were introduced as replacements for file cabinets, it was considered sufficient to retrieve an image in a minute or less when compared to the minutes, hours, or days required to retrieve a physical file. Today, many businesses use DISs for mission critical applications where even a few seconds delay can cause productivity loss and customer dissatisfaction. It is not uncommon to find requirements for page flip rates of one to two seconds.

Clearly, this level of performance requires careful analysis and design. Developers must study retrieval patterns to determine how the images will be used. System configurations are affected by caching images on faster peripherals, such as magnetic disk, for retrieval. Faster processors with more memory and decompression boards may be required. The network topology and protocol may be impacted. And the options of standalone optical drives versus optical jukeboxes must be considered.

Backfile Conversion

Many organizations start their plans for implementing a DIS with the idea of converting all existing paper files.

If this is the case, as an adjunct to document analysis, the developer will need to identify the volume and condition of the old files. As part of the analysis, the cost of capturing the old files is calculated. This includes not only the amount of storage required, but the labor cost of scanning and indexing. Frequently, the user finds that these costs outweigh the benefit of capturing the old files.

If the user accepts the costs, the logistics for accomplishing the task must be considered in the implementation plan. The amount of time required may involve weeks or months of scanning. Staging the backfile conversion to coincide with the availability of a working system is required. Two options are frequently used for backfile conversion. Both require careful planning and management.

1. The design and coding of the software, as well as the design of the configuration and the integration of the hardware and software is largely complete before the process can begin.
2. The conversion is carried out independently with only enough indexing to allow the images to be imported into the final database. Full indexing is completed after the application is substantially in place.

Versioning and Image Editing

As part of the image analysis, the developer must identify if image overlays and versioning are required, as they are for redacting (editing) applications where the original image must remain unchanged. In other instances, the image management system must support editing of the image file. These requirements will affect the choice of the image software, as well as the database design.

Image Retrieval

Developers are accustomed to studying the way that users retrieve data to determine how to structure database tables and indexes. With a DIS, the developer must analyze the information embedded in the document image and determine how the user can best retrieve that document when needed. This analysis impacts the database index design. Additionally, the developer must assess how the index is to be populated. If automated indexing is required, an OCR or ICR software package may be required. If full-text indexing is needed, then OCR will be included.

The indexing strategy is also impacted if the DIS must tie into an existing legacy system, frequently mainframe-based. In this case, the developer must determine if the index for the document is to be stored on the mainframe, on the imaging system, or on both. This introduces data

location issues related to replication and integrity of data.

Workflow

While workflow is beginning to be included in the design of traditional data processing systems, it has more commonly been used with DISs. In early implementations, it was not much more than a simple routing system, sending the document from worker to worker for processing. Today, workflow products include facilities to support sequential and parallel workflows. Some systems allow administrators to define conditional flows and automatic retrievals. Dynamic or manual workload balancing is another important feature, along with use of priorities to schedule some work in preference to other less critical processes. These features come into play as the developer analyzes existing document flows and designs new processes.

Importance of People

The DIS issues presented above require specific knowledge of the technology and the implications of implementing a solution. However, without a thorough analysis, which includes a complete and accurate understanding of the requirements, the chances of a successful implementation are less likely. The people in an organization are the most important resource for

achieving an accurate definition of the system requirements.

During the analysis of requirements for automating a manufacturing process, for example, the individual worker on the shop floor is likely to be able to provide the system developer more information about the intricacies of the process than the shop floor supervisor. The same is true of the people who handle the paper in an organization. The manager may know the company policies and the processes based on a company manual, but the clerk who processes the paper knows where the problems exist, what data is present or missing at various stages in the process, and how the company policies have been adapted to the reality of getting the work done. On the other hand, the clerk probably views the system as a mechanism for entering and accessing data without understanding the complexities of the system internals. And management approval for the new process is necessary to ensure adequate backing for the new system.

It is important to recognize that most methodologies do not include much, if any, emphasis on human factors in the analysis and design of a system. This aspect of implementing a DIS can lead to dismal failure, if not properly addressed. The same worker who is handling the paper will be spending their day in front of a computer terminal after the DIS is installed. There will be none of

the normal breaks to track down a file. While management views this as a way to improve productivity, the net effect on the end user and the work processes can be very detrimental. Repetitive stress injuries may increase because there are no breaks in the routine. Water cooler conversations may be reduced, but the transfer of information between end users may also be stymied.

There must be a mechanism for analyzing the human factors aspect of a business, determining what is critical to the work being done, and designing appropriate mechanisms into the new system to accommodate them, as necessary.

When the automation of a manufacturing process is being considered, the first thing that is likely to be done is a survey of existing technology for suitability to the task. Document imaging is a relatively new technology and many system developers, as well as the end users, are not familiar with its capabilities. A primary challenge, therefore, is to educate everyone who will be involved in the project: end users, middle managers, and executives. These individuals must become "owners" of the system so that they will be willing to commit themselves to its success. It is important for the system developers to understand the contribution that users can make to the analysis and design of a DIS that will meet the needs of the organization.

End Users

Of all the people involved with a DIS, the end users are undoubtedly the most heavily impacted by the new technology. Most people have become accustomed to the idea of using a computer in some way to help them get their job done. However, with document imaging, the way in which they do their job will become fundamentally different (Cinnamon and Nees 1991).

Currently, they might pull a file of papers and flip through them to find the one they want. Perhaps they stick a ruler, pen, or piece of paper into the file at various places where they find specific information. If the phone rings, they may leave that file, pull another file, and respond to the caller's request for information before returning to the original file. They may have a stack of several folders on their desk that are in various stages of processing. With the new DIS, there will be no paper files. Images of the documents will be called up on a high-resolution monitor. How will these people deal with the lack of a physical file? How will they mark their place as they flip through a file? How will they even flip through a file? How will they handle multiple files or multiple requests?

The challenge for the system developer is very complex, because there must be an understanding of much more than just the types of data stored on the paper in the

files. How people work with the documents, as well as the relationships between various types of documents, becomes very important. If the documents are not presented in a usable way, the end result is likely to be an "informal" file, printed copies of the documents that the user needs to complete a task. Rather than reducing paper, additional paper is generated. So the system developer must analyze not only the information on the paper, but also the way the paper and the information are retrieved and used. The only way to do this is to involve the end user directly in the systems analysis and design process. Therefore, the selected methodology must support this type of interaction.

An additional factor that relates directly to the end user of a DIS is the environmental aspect of the job as it will be performed after the system is installed. Users who used to worry about paper cuts and falling file cabinets will now worry about radiation from the monitors, repetitive stress injuries from using the keyboard, and eye strain or headaches from looking at a screen all day. The systems analysis and design methodology must allow the human factors issues to be taken into account and resolved in an acceptable manner.

Middle Managers

This group of people may represent the biggest challenge to the system developer, especially in a records

management department. They will have established an organization that is rarely singled out, except when someone cannot locate the papers they need to solve some high priority organizational need (Trammell 1992). Yet they, similar to a librarian, have built an organization and a system for handling what may be massive volumes of paper documents. They may have to deal with microfilming of documents, which includes the complexities and safety issues related to film processing.

These people are typically very protective of their territory and may feel the new technology is an invasion. They are, however, critical to the success of the new system since they are the repositories of key information about the way the organization revolves around the documents within their purview. The DIS methodology must be responsive to the concerns of middle managers, especially those responsible for records management within the organization.

Executives

Setting realistic expectations of senior management is an important part of the system development process for implementing a DIS. Vendors of imaging systems are very eager to describe the fantastic returns on investment (ROIs), improvements in productivity, and staff reductions that can occur with a DIS. Rarely do they set an

appropriate level of understanding relative to the amount of work that must be done to achieve those types of results. Any methodology used for this type of system must include a cycle of interaction at the executive level to identify realistic goals for the system. While it is true that there is the opportunity to achieve dramatic changes, not all of those changes come quickly or easily. If a project is on the critical path to meet organizational goals, or if management becomes overly anxious to see a positive ROI on the books, the system design is likely to suffer from being rushed into production (Trammell 1992). However, if a clear understanding of the process is achieved and expectations are set at a realistic level, it is reasonable to expect management to accept the fact that the system will have to evolve over a period of time to reach its full potential.

Summary

As system developers face the challenge of implementing a DIS, it is clear that more is involved in the process than installing the technology. These systems directly impact the users in ways that may not initially be identified. They also have the potential to impact the entire organization and the way it does business. There are additional complexities that must be handled relative to capacity, performance, and workflow, among others. If

implemented well, a DIS can facilitate the addition of new services, a reduction in overhead expenses, and an increase in customer satisfaction. In the competitive environment businesses encounter today, it may be the edge that keeps them ahead of the competition.

CHAPTER 4. INTERVIEWS AND OBSERVATIONS

AIIM publishes a magazine, Inform, which contains articles related to document imaging systems. In January 1993, they included an article by Thornton A. May relating the results of a survey conducted by Tenex Consulting, Inc. This survey of professionals who have been DIS project managers showed some disturbing results, especially for anyone considering a career in this specialty. Twenty-five percent of the respondents reported that their careers had suffered severe setbacks as a result of trying to implement a DIS. Another 17 percent indicated that their work was not completed and they were "still trying" to get the systems to be fully accepted by the users. Twenty percent had abandoned the DIS technology and were trying to solve the original problem using a different technology. And 33 percent had performed further reengineering of the system than was initially anticipated before they were able to gain user and management acceptance.

One problem identified by May (1993) is that expectations are often misdirected. Executive level managers become disillusioned when the cost of the system exceeds their perception of its worth. While May concentrates on the tendency of understating costs and overstating benefits of a DIS, which are valid concerns, it is feasible that not designing a system to truly meet

business and user needs is a contributing factor in the seemingly high failure rate.

To ensure that the proposed DIS methodology is effective, it is important to examine the factors that contribute to the successes and failures of implementing DISs. This chapter presents the results of interviews with individuals who relate their real world experiences with DISs. The problems and benefits these professionals encountered with various implementation approaches are important considerations for developing the DIS methodology.

DIS Interviewees

The candidates interviewed are computer professionals who have been involved in analysis and design of DISs for two to seven years. They are all currently active in the field and have had a wide range of experience with image-based systems. The composite of those interviewed include the following traits:

- One individual first became involved in imaging technology from the perspective of satellite imagery. The other DIS professionals got involved in document imagery as an extension of their work with data processing systems. For example, one person started working as a programmer for data processing systems and moved into document imaging in the late 1980s.

Another was involved in a variety of programming and systems integration projects ranging from financial to office automation applications.

- The projects were largely for government offices or agencies. They ranged from $25,000 to multi-million dollar systems.
- Each of the individuals interviewed worked for an imaging system vendor or systems integrator when they implemented the DISs.

Together they bring a broad range of thoughts and experiences about the processes they went through.

DIS Experiences

During the interviews, the topics covered included both DIS-specific and general data processing related issues and concerns. They ranged from general approaches for introducing DIS technology to the fine points of application design and implementation. The discussion that follows represents the highlights of the information gathered during the interviews.

Methodologies

Each of the individuals interviewed had adapted tools and methods from earlier career experiences to their own approach for the analysis and design of DISs. One

individual stated that the scientific method is an appropriate approach in the absence of any formal system development methodology. Document processing depicted via programming-like flowcharts is also used as a tool to depict current and proposed document flows, rather than data flows. A formal methodology is available to project managers within one organization. This methodology is an adaptation of the Information Engineering approach. As a matter of practice, the project managers adapt the phases and deliverables to the system being implemented and the requirements of the customers.

Based on the interviews, the success or failure of the DIS did not appear to be related to the use of a standard methodology for implementing the system. In many cases, the users and management resisted the use of a full system development methodology for their system implementation. A number of reasons were given, some resulting from a complexity of the process involved, but primarily attributable to the desire for a shortened development cycle.

Users rarely understand the long term benefits accrued from the use of a methodology. They see the amount of time spent on analysis and design, while they wait for their system to be implemented, and they question why the developer cannot simply start building the system based on their statement of needs. Historically, users have seen

information systems (IS) organizations depart with the requirements specification and return with systems that do not meet user expectations, are later than promised, and cost more than originally projected.

Users tend to reject the "trust me" attitude that has pervaded much of the data processing world since the days of the glass houses for mainframe systems. Users with PCs on their desks are better prepared to be part of the analysis and design of a DIS than their earlier counterparts.

Diagramming Tools and Techniques

All interviewees agreed that the use of diagrams was essential to good communications between the system developer and the user. Complex diagrams, however, may intimidate the users and prevent them from understanding what the developer is trying to convey. The diagrams generated during analysis depict the current system and users are expected to validate the analysis based on the diagrams. If the users cannot understand the diagrams, they sometimes approve the diagrams, whether correct or not, just to keep things moving. The belief seems to be that they can correct it later if it doesn't come out right. The consensus of those interviewed is that the diagramming method must be easy to teach to the end users.

One of the problems with the diagramming techniques used with many methodologies is the lack of a scheme for depicting sequencing of transactions. Adaptations used by developers include adding sequence numbers to the process flows and the use of flowcharting to depict document processing flows. While sequencing may not be critical for a simple storage and retrieval system, it is essential when examining a workflow application.

Another problem cited with many analysis and design tools and techniques is the lack of support for graphical user interfaces (GUIs). Application design and development previously involved a single focus. Once an application was invoked, the user entered data, performed queries, and generated reports. When they wanted to use a different application, they terminated the first and invoked the second. Today, users are able to have many applications active at one time, switch focus between those applications, and, in fact, have the applications interact. In a DIS, this is often the case when there is interaction with an existing mainframe application. As a result, new tools and techniques are needed to appropriately handle the new implementation options and to be able to clearly depict them for users.

User Involvement and Human Factors

A frequently cited reason for a DIS system failure is lack of end user involvement in the development process. An implementation where the IS organization interprets user needs adds another opportunity for the true requirements to be translated incorrectly. In many cases, the IS group feels that it is unnecessary to involve the users in the system implementation.

At least two types of logic may lead to the isolation of the user. First, the IS organization may have the attitude that IS staff, the computer professionals, know better than the users what should be implemented. A second reason voiced by IS staff is that the users are far too busy keeping up with their day-to-day work and must be protected from getting involved in the system development. However, the end result is often that needed functionality is missing. It is also possible that the IS group will perceive a need for capabilities that might be considered gold-plating and unnecessary by the end users.

Another situation that often yields an unusable system is when a request for proposal (RFP) dictates the implementation without allowing the winning vendor an opportunity to conduct an analysis phase. Too often the RFP is unclear or open to multiple interpretations. If the vendor implements the system based only on the RFP, DIS project managers reported problems implementing the system.

The bottom line is that, regardless of the methodology used, the DIS development process must involve users from all parts of the organization in the analysis and design if there is to be a reasonable chance of success.

Summary

On the basis of the interviews and various readings, there are many reasons that traditional system development methodologies are not used. Among those reasons are several common to most users.

- The methodology is too complex and users cannot understand the value that they will derive from it.
- The deliverables are too confusing to allow the user to understand the analysis or design.
- User input into the design phase is typically too late to allow substantive changes.
- Users do not trust system developers to deliver systems that meet their needs on a timely basis and within budget.

Some of the changes in the types of applications being developed have led to incompatibility of traditional system development methodologies. The multi-focused applications in use today are not well supported by the structured approach to system development, but are a better fit using OO design tools and techniques. This type of problem is

not unique to DISs and is not addressed as a separate consideration in the proposed methodology.

The concerns about the complexity of the methodology and diagrams also are applicable to both DISs and traditional information processing systems. However, the issue of communication between developers and users is considered to be essential. Therefore, the proposed DIS methodology must provide an easily understood method of communicating analysis and design specifications between the different groups.

Finally, user involvement in the system development process is considered a priority in the proposed methodology. Again, this is a concern in all types of system development today. It is considered essential for DIS implementation due to the human factors impact that the DIS has on the end user.

CHAPTER 5. DEVELOPMENT OF A METHODOLOGY FOR DISs

Chapter 1 presents the hypothesis for this research project raising for consideration the idea that an analysis and design methodology for DISs must emphasize three topics: 1) understanding business processes, 2) human factors and ergonomics, and 3) use of prototype or pilot systems. Chapter 3 includes a discussion of DIS-unique capabilities which require specific activities during analysis and design. Chapter 4 provides anecdotal evidence of the difficulties system developers frequently encounter with actual DIS projects.

This research project is based on the belief that traditional methodologies, such as structured systems analysis and design, do not adequately address these topics during system development. The methodology proposed here addresses DIS requirements, where appropriate, throughout the system development life cycle.

Proposed DIS Methodology

Figure 9 represents the types of activities which might be carried out in a DIS development life cycle. The proposed DIS methodology supports parallel activities for five aspects of system development:

- training
- analysis, design, and implementation

- testing
- documentation
- cost/benefit analysis

The Mills spiral is used as the system development process framework for the proposed DIS methodology because of its ability to adapt future activities to the outcome of current work. As an activity is completed, the past activity is evaluated and the next activity is selected. Activities can be repeated, moved up or down the spirals, spun off into parallel spirals, and added or deleted as appropriate. The vertical lines in the spirals only loosely represent time, since activities can be moved around. The dotted lines across the spirals indicate suggested phases within the development process.

Due to the complexity of implementing DISs and the uniqueness of every organization, each system is likely to require a customized sequence of activities and spirals. Since the Mills spiral model is based on the concept of a growing model, it provides a dynamic model for use with DISs.

Within this framework, the Box Structure Methodology (BSM) supports analysis of the current and proposed systems, as well as design of the proposed system. BSM allows the developer to depict the sequencing of processes, an essential element for analysis of document flow. It

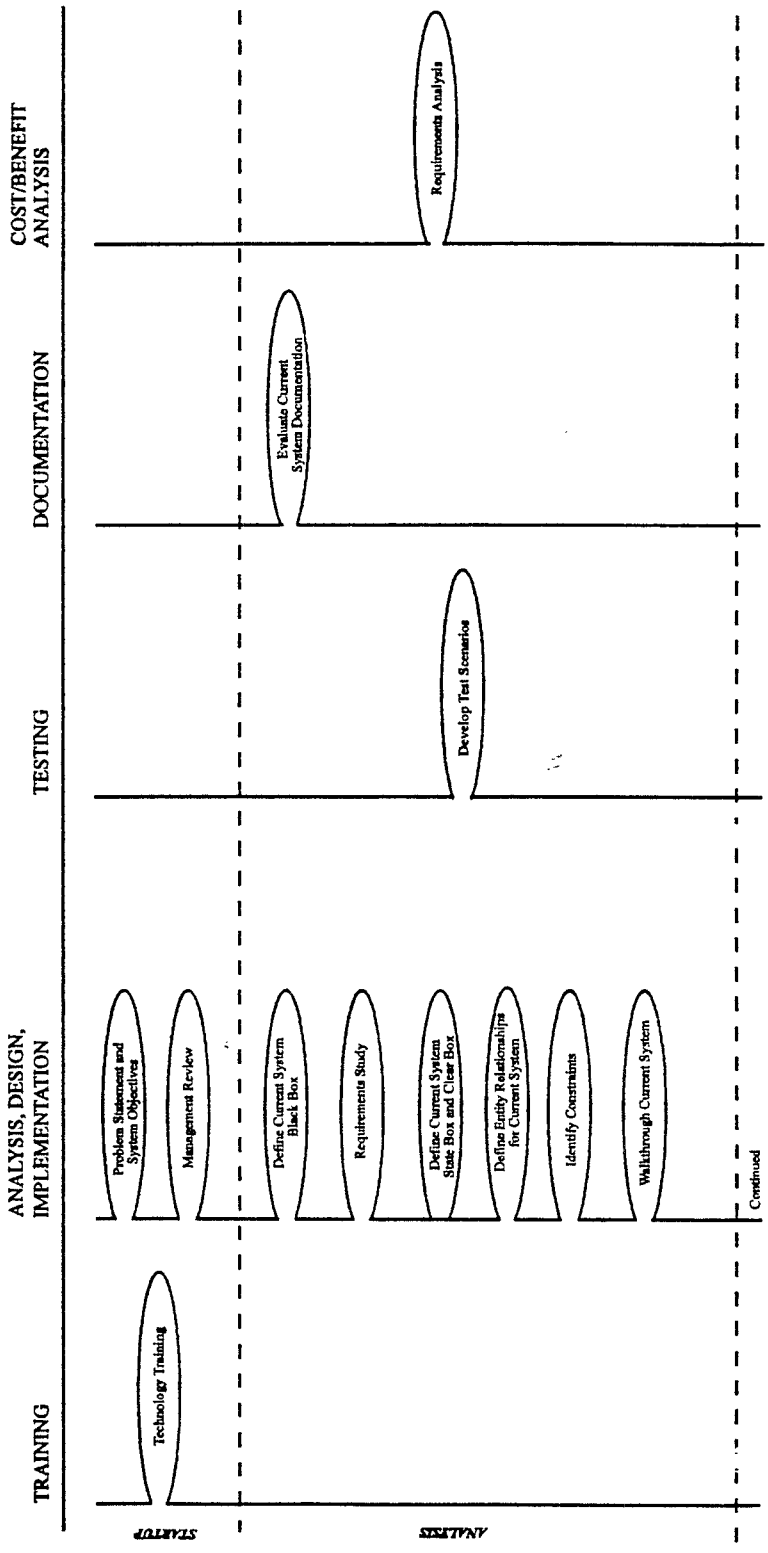


Figure 9. Proposed DIS Methodology

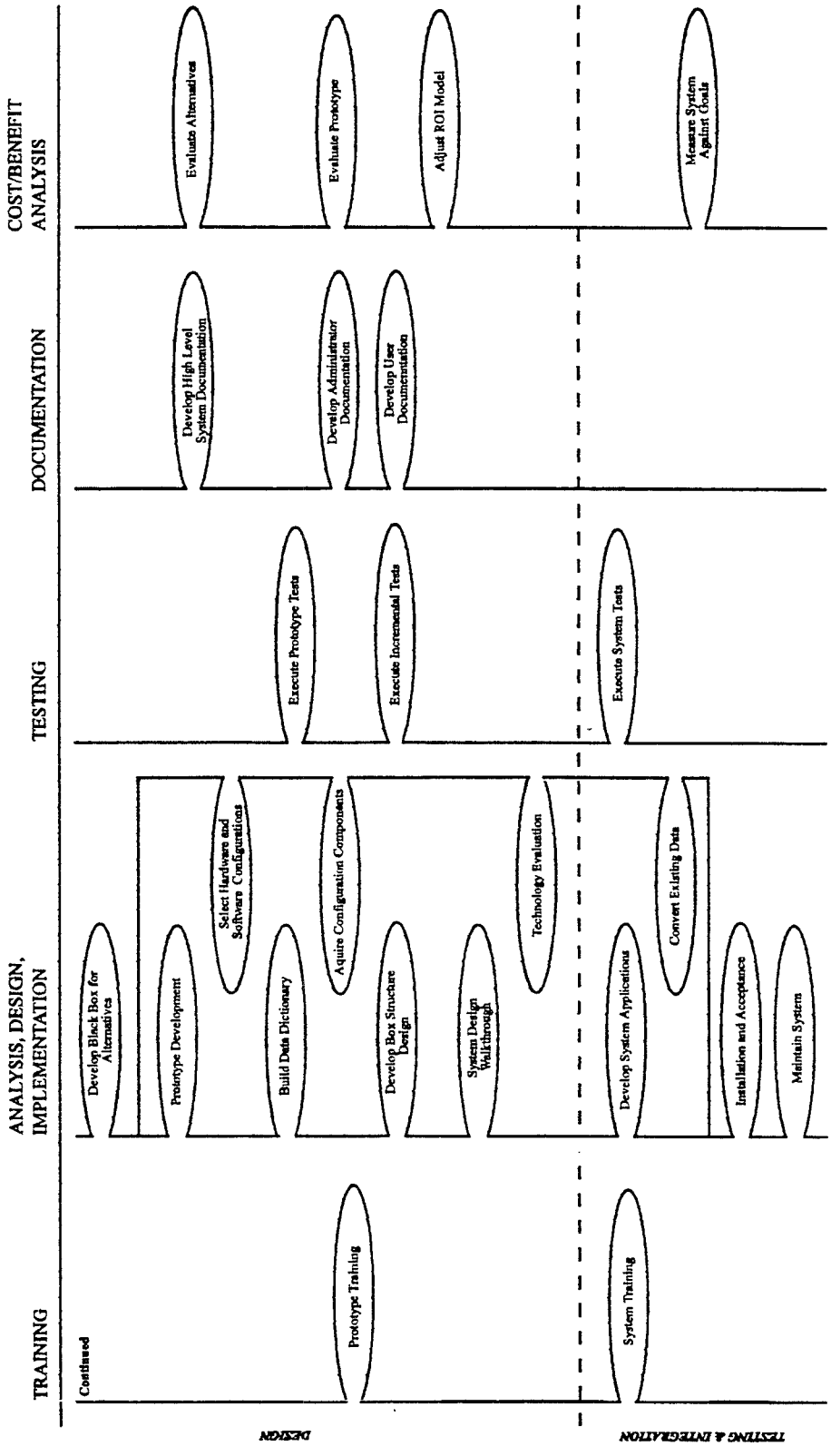


Figure 9 (continued). Proposed DIS Methodology

also provides a mechanism for identifying process problems in the current system and validating the design of the new system.

Upon reviewing the proposed methodology, the reader may wonder why this approach would be used exclusively for DISs. In fact, it is general in nature and could support a wide variety of data processing systems. What this methodology does offer for a DIS is the ability to accommodate the unique features presented in Chapter 3, as well as the specific topics emphasized in Chapter 1. The following discussion points specify how the DIS implementation details are supported by the proposed methodology. In cases where the activity does not address DIS-unique requirements, the reader is directed to seek references in the current literature for typical usage.

Training

Various levels of training are necessary for new DISs. In organizations where imaging is not currently used, the training should begin early in the development life cycle and should include all levels of management and staff.

Technology Training -- A successful DIS implementation can be achieved by ensuring that expectations are correctly set and that everyone is comfortable with the technology. With office automation and data

processing systems, many people have a reasonable level of confidence and comfort. They know they can enter the data into the system or create a spreadsheet with predictable results. There is a level of assurance that the data and files will not be lost. However, people feel very nervous about scanning a document, saving it to optical media, and destroying the paper document. The more important the piece of paper, the greater the uneasiness. The purpose of technology training is two-fold. First, it is to introduce new hardware and software capabilities to the users. In addition, technology training is used to raise the users' level of confidence that not only is the document permanently recorded but that it can easily be retrieved and displayed.

This period of training also serves as an excellent opportunity to experiment with a prototype of the user interface. Early training on a system using mock system screens and functionality allows developers to observe the way documents are retrieved and used.

Prototype Training -- Many traditional methodologies have been modified to support prototyping, but they frequently are not flexible enough to accommodate multiple iterations within analysis and design.

Since, the proposed DIS methodology introduces the prototype early in the process, users can provide feedback at a stage when it will be most useful.

Once the development data dictionary is underway, it can be used to develop prototype screens and possibly a working prototype system. The use of this prototype enhances user input into the design and provides additional hands-on experience prior to implementation. Importantly, user training on the prototype can lead to early identification of human factors design issues. As noted earlier, user participation also enhances the possibility of user acceptance.

System Training -- This training is traditionally conducted for new systems and is not unique to implementing DISs.

Analysis, Design, and Implementation

Distinct milestones mark the end of one phase of DIS system development and the beginning of the next, but they do not restrict the project manager from allowing overlap between phases. When incremental development is used, a high level analysis and design can be completed and development can start while the next level of analysis is being conducted. If this is permitted, there will be

parallel spirals within this area of the DIS methodology. Steps can also be repeated as necessary. It is critical that all levels of users be involved in these activities.

Problem Statement and System Objectives -- This activity is not unique to a DIS. The developer must, however, keep in mind that there are unique features of a DIS that can influence the ability to develop a system that meets all objectives.

Define Current System Black Box -- As with traditional data processing systems, the DIS life cycle requires an understanding of the current system. It is at this stage that business processes and transactions are identified. This analysis is useful in defining the workflow for the new system.

Requirements Analysis -- Developing a DIS requires an analysis of documents, including not only volumes, but document types and document handling. Initial image capacity analysis is performed to identify a baseline amount of magnetic and optical disk. Document content is examined and the effect of image compression options on image file sizes are estimated. This is done early, to allow user requirements to be re-evaluated if the volumes exceed either the capacity of

the system or the user's budget. Requirements for image editing are also identified at this step. The results of the analysis are used extensively in developing alternatives and in selecting the hardware and software for the new system. Additionally, a study of the image retrieval patterns provides important input during the design steps when alternatives for document index, image storage, and image caching are being developed and evaluated.

Develop Current System State Box and Clear Box -- Again, the DIS developer relies on the same activities that are used with other data processing systems. For a DIS, the developer can use the clear box definition of the current system to verify that the current system satisfies the requirements and can begin to determine how the business processes might be re-engineered to provide more efficiency (Hammer 1990). Workflow analysis for a DIS is supported during this step. The verification and validation steps of the BSM provide an excellent mechanism for validating that business requirements are met and identifying areas where the business processes can be enhanced. As a part of this activity, specific performance requirements should be identified and quantified.

Define Entity Relationships for Current System -- If the current system is automated, rather than manual, the DIS developer must consider whether images can be added without negative impact. As the entity relationships are recorded, the developer may find that the existing system presents constraints on the design alternatives that are feasible.

Identify Constraints -- The DIS developer must be aware of the impact of standards used within DISs. Especially when developing systems for federal government agencies, the standards required may place constraints on the hardware and software technology that can be used. Standards may also impact the platform that is available for the user interface.

Walkthrough Current System -- As with all systems development, walkthroughs constitute a validation of the work already completed. User participation in walkthroughs is particularly important for DISs implementations, due to the significant human factors impact on the system users.

Develop Black Box of Alternatives -- Many of the unique DIS requirements are supported in this step. As a result of the document analysis conducted

earlier, the developer must perform a trade-off analysis of the compression algorithm to be used. Factors in this analysis include:

- the type of documents
- the most effective algorithm for those types of documents
- the performance implications of the compression and decompression processing
- the magnetic and optical disk capacity required to store images compressed using the alternative algorithms
- the cost of compression/decompression hardware and software

At this stage, the developer analyzes the options for document indexing and location of the indices, as well as for the inclusion of OCR, ICR, or bar code technologies. Additional factors of image storage and imaging caching can also be evaluated.

Alternative document capture strategies are considered at a high level during this step, but more detailed analysis may have to be conducted during a subsequent step. Throughput analysis for each alternative should be conducted to ensure that the system can capture the document volumes within prescribed timeframes.

Prototype Development -- Based on the results of the analysis steps and the preliminary design alternatives, a working prototype of the system is developed. If a prototype is included in the Technology Training step, user comments from that activity are incorporated into the prototype design, as appropriate. The prototype is used in the Prototype Training activity to provide additional user feedback about the design and functionality of the user interface. The individuals responsible for testing also use the prototype in the Execute Prototype Tests activity.

Build Data Dictionary -- The process of developing the database is not unique to DISs.

Develop Box Structure Design -- As with other automated systems, this step provides the detailed design of the new system. Nothing new for a DIS implementation is introduced in this activity. However, each DIS-specific factor should receive the attention of the developer for detailed design.

System Design Walkthrough -- As mentioned for the current system walkthrough, this step is critical to the success of all systems, not just DISs.

Select Hardware and Software Configurations -- As with all automated systems, hardware and software components are selected. The DIS adds a level of complexity to the process as the technology may be unfamiliar to the user or the user may have pre-formed ideas of the capabilities of various components. While these additional DIS components increase the complexity of the task, the processes for carrying out this step are not unique to DISs.

Acquire Configuration Components -- This step too follows standard procedures. In the implementation of a DIS, however, components may have to be acquired ahead of the schedule normally used for automated systems. This may be necessary to facilitate prototyping or backfile conversion.

Technology Evaluation -- The complexity of integrating imaging components and dealing with the additional load that images place on a system should not be underestimated. While the same types of installation, integration, and troubleshooting skills are used in implementing both DISs and other automated systems, this activity requires additional attention for a DIS. Unless all products were successfully integrated for a previous project, it is necessary to prove that they

can be made to work together as required. For example, moving images from a scanner to a separate OCR engine may not be as simple as transferring the file due to differences in the image file header or compression techniques used by the two products.

Develop System Applications -- Programmers must be aware of the implications of using images and image management software. The activity of application development is, however, not unique for DISs.

Convert Existing Data -- As mentioned in Chapter 3, many organizations do not understand the cost in terms of time and resources required for backfile conversion of documents. At this step, the conversion effort is undertaken. Depending on the strategy selected and the volume of material to be backfiled, this step may require weeks, months, or even years. In a DIS implementation, scheduling appropriate amounts of time for this process can be more critical than for other automated systems implementations.

Installation and Acceptance -- This step is a part of the implementation of any automated system and is not unique to a DIS.

Maintain System -- All systems require planning for maintaining the system. This is an ongoing activity to provide support for the system once it is in production.

Testing

The Cleanroom methodology specifies that testing is a concurrent activity to system development. The same approach is recommended for the proposed DIS methodology. This testing approach is not unique to DIS implementations, but it is critical to ensure that requirements are satisfied.

Develop Test Scenarios -- Responsibility for testing will ideally be given to an independent test team. This team will develop the initial test scenarios based on the requirements and critical success factors identified in the Problem Statement and System Objectives activity. Tracking requirements from the originating documents ensures that something is not inadvertently overlooked as the system evolves.

Execute Prototype Tests -- At this step, the testers can apply the test scenarios to a working prototype. With this early testing of the system design, there is

an opportunity to validate that the system remains on track with the initial requirements.

Incremental Tests -- Testers validate that system components and subsystems function as specified. If the rules of referential transparency are observed, new code modules can be tested and added to the working prototype of the system without introducing errors and requiring earlier tests to be repeated.

Execute System Tests -- Full system tests are a parallel activity during Installation and Acceptance. During this activity, the developers satisfy the users that the system functions as required.

Readers interested in pursuing additional information on testing may find texts by Mosley (1993), Hetzel (1988), and Myers (1979) helpful.

Documentation

Often left to the end of the development life cycle, the proposed DIS methodology recommends that the documentation phase be a parallel activity to analysis and design. This is not unique to a DIS implementation. Documentation activities include:

- Evaluate current system documentation

- Develop high level system documentation
- Develop user documentation
- Develop system administrator documentation

Cost/Benefit Analysis

Adequate monitoring of project costs, both current and projected, is essential to maintaining cost control. This is an ongoing task, with specific checkpoints, as identified below.

Requirements Analysis -- During the requirements study for a DIS, the documents to be stored in the system are studied and estimates are made of the magnetic and optical capacity requirements. When these calculations are complete, it is important to reevaluate the proposed system to determine if the plans are still feasible. If the capacity required for the documents raises the cost of the system beyond the original forecasts, it may be necessary to adjust the requirements to fit the budget or increase the budget. This could include a trade-off analysis of the costs and benefits of alternative document storage strategies.

Evaluate alternatives -- At this step, regardless of the type of system, the system developers should

examine all alternatives to determine which provides the best value to the users. This activity is not specific to DISs.

Evaluate prototype -- Changes to the system design resulting from use of the prototype during training and testing can have an impact on the cost of the system. In cases where new products or alternative programming approaches are required to satisfy the requests for change, a trade-off analysis may have to be conducted. This evaluation is used to determine what can be done within the financial and schedule limitations for the system. If the changes are necessary, additional resources may have to be budgeted. This activity is not unique to DISs. However, since the proposed DIS methodology places such emphasis on the prototyping, it is more likely to be needed.

Adjust ROI model -- Once an implementation alternative is chosen, the return on investment model from any earlier feasibility study should be modified. This is the model by which the system's cost effectiveness will be measured.

Measure system against goals -- Periodically, after a system is in production, measurements should be taken to determine if the projected benefits are being realized. This activity is essential for all systems. As time passes, these measurements will show when the system is no longer cost effective and should be replaced.

Summary

The proposed analysis and design methodology supports the DIS-specific issues and factors mentioned in Chapters 1 and 3 and emphasized in Chapter 4. It provides flexibility for adapting the steps within the phases to fit the requirements of a particular implementation. If users are familiar with the technology, less emphasis can be placed on the Technology Training. The additional considerations for image capture, storage, and retrieval are accommodated within the phases of the proposed methodology. Users are included throughout the system development process, from analysis of the current system through verification that the new design meets user requirements.

CHAPTER 6. EVALUATION OF PROPOSED DIS METHODOLOGY

This chapter presents a case study of the proposed DIS methodology, focusing on activities that address three of the DIS issues raised in Chapter 3:

- Document analysis
- Image compression options
- User interface design

It is critical that the methodology support these three activities to ensure that the DIS implementation is successful. This work occurs early in the system development life cycle and, therefore, impacts many of the activities that take place later. The proposed methodology is flexible enough to adapt to the unique characteristics of a DIS, while providing control for the flows of information out of these activities and into those activities that follow.

This project begins with the Analysis, Design, and Implementation spiral, defining the system requirements and objectives. In other situations, the process could begin with another spiral. An organization with no DIS experience could start with Technology Training under the Training spiral to allow users at all levels to become familiar with the technology before finalizing the system objectives. By educating the users about document imaging,

they will be better prepared to participate in the analysis and design activities. The staff at the company participating in this case study is very familiar with DISs.

The Case

A request has been made for a system to maintain a library of readily available technical specification documents for a systems integration company. The engineers at that company are tasked with finding components that meet their customer's requirements. To prepare for product identification and selection, the engineers acquire a large quantity of marketing and technical literature. This product information is also used as reference material during proposal development.

At present, each engineer keeps a separate, manual filing system. Since the number of available products is quite large, it creates a burden on each engineer to maintain their files so that the material is easily retrieved when needed. Engineering managers feel that a centralized, well-controlled DIS will have several benefits:

- A central repository for technical documentation will provide all engineers with access to the same, broad range of information. Frequently, one engineer will discover a new product which may be of interest to

other engineers, but the current methods make sharing difficult. As a result, excellent products that meet technical, as well as price/performance criteria, are sometimes overlooked.

- By using an on-line database, searches across product categories or vendors will be facilitated. It is currently impossible to maintain a manual filing system that supports both types of searches equally well.
- By using a DIS and storing electronic copies of the documentation, many engineers will have access to the specification sheets simultaneously. Currently, to do this they must make copies of the documents.
- A DIS will eliminate the problem of missing or misfiled documentation, since the images are not actually removed from the system when they are viewed or printed.
- As new literature is received, there is a tendency for the engineers to add the new documents without removing any of the obsolete material. The process of deleting out-of-date and superceded documents will be facilitated by the DIS.

The originally stated objectives are that the system be easy to use; not take a lot of time to maintain; use

existing hardware and software, if possible; and utilize few technical resources during development.

A simple database would provide some benefit to the company. However, with such a large number of disparate components being used in customer configurations, the number of data elements that would have to be entered to facilitate retrieval of product information would make the system too burdensome to update on a regular basis. By implementing a DIS, the data entry can be kept to a minimum, just enough to allow the engineer to identify a range of products. Further product qualification is done by providing the engineer with the capability to view the specification sheets stored as images within the DIS.

Problem Statement and System Objectives

Input:	Original Problem Statement Original System Objectives
Output:	Refined Problem Statement Measurable System Objectives

The first task for this step is to refine the problem statement and establish the system objectives in terms of measurable critical success factors. In addition to flowing directly into the Management Review step, the output of this step provides the baseline requirements used in the Develop Test Scenarios and Installation Acceptance steps of the methodology. The criteria must be well

defined so the tests truly reflect the way the system will be used and judged by the engineers. The output is also used during the Define Current System Black Box to focus the analysis on the problem to be solved. During the Develop Black Box for Alternatives step, these requirements are used to validate that the design choices support the users' needs.

For this case study, the original problem statement is accepted as written, but the objectives are modified so that they are measurable.

- Ease of use: Must be able to train a new user to scan and index documents in less than 1/2 day. Must be able to train a new user to retrieve and print documents of interest in less than 1/2 day.
- Time to maintain: Must be able to scan and index an average of 30 documents per day in less than 2 hours.
- Hardware and software: Other than optical devices and media, use existing hardware and software currently available for DIS development, support, and demonstration.
- Development resources: Allow internal staff to design and develop the system without interfering with existing customer deliveries and new business support.

Management Review

Input:	Refined Problem Statement Measurable System Objectives
Output:	Approved Problem Statement Approved System Objectives

The management team must agree to the specifics of the problem statement and the system objectives which are the product of the previous activity. This approval must be gained to ensure that the delivered system will have defined criteria against which it can be measured and judged. One benefit of using the spiral paradigm for the proposed DIS methodology is that it allows the parts of the input that are approved to be forwarded to the next process, Define Current System Black Box, while the parts that are not accepted are resubmitted to the previous process for further refinement.

Define Current System

Input:	Approved Problem Statement User Interviews Current System Documentation
Output:	Current Logical System Description

Business processes and document handling are analyzed in this step. The Approved Problem Statement document is used to help the developer focus on the problem(s) to be solved. The system developer conducts interviews and

gathers information from users at all levels. End users explain how they do their jobs and provide details about what data they need at each step. Middle managers and supervisors provide information about how they control the work that flows into and out of their area. Upper management supplies information about goals for the organization and what work they believe is being done. Evaluation of Current System Documentation is a parallel activity under the Documentation spiral from which information about the current system functionality is drawn.

The accumulation of all of the interviews is analyzed by the system developer and reviewed with the users and managers to ensure that the work has been captured accurately. The output of this activity forms the basis for business process reengineering that is done in the Develop Black Box for Alternatives and Develop Box Structure Design activities.

Requirements Study

Input:	Approved Problem Statement Approved System Objectives User Interviews Paper-based Documents Current Logical System Description
Output:	Description of Documents Image Capacity Projections

Two DIS-specific activities are conducted during this step: Document Analysis and Image Capacity Analysis. Both of these analyses are conducted as part of this thesis and are presented below. The problem statement and system objectives, output from the Management Review step, are used as input. Users supply information about what constitutes a document, how it is handled, and what document data are used. The system developer examines the documents to identify the characteristics pertinent to document capture, storage, and output. Users may provide estimates of the volumes to be used for image capacity analysis.

The output from this step is used in other steps of the proposed DIS methodology. Document analysis provides preliminary data for developing a prototype user interface. The results of the image capacity analysis are used to determine feasible alternatives and are directly input into the hardware and software selection. The output also flows into the Develop Test Scenarios process to provide additional understanding of user requirements that should

be part of the test suite. The Identify Constraints step also makes use of the output from this step. Document content, usage requirements, and image compression characteristics can all impose constraints on the system under development.

A parallel activity in the Cost/Benefit Analysis spiral reviews the costs associated with the image capacity analysis and evaluates the projected benefits. Adjustments in the number or types of images to be stored or the image compression algorithm can be made if the analysis shows that costs are out of line with the budget.

Document Analysis

One of the early activities in the Requirements Study step is Document Analysis. In Figure 10, the activity loop for Requirements Study is highlighted. The Requirements Analysis loop is also highlighted to indicate that there are Cost/Benefit implications for the findings of the document analysis work.

A stack of technical material was examined and categorized based on a number of relevant criteria, including form size, ink color, paper color, and binding. The material for the Tech Library consists of product information received through the mail, during meetings with vendors, and from technical exhibitions and seminars.

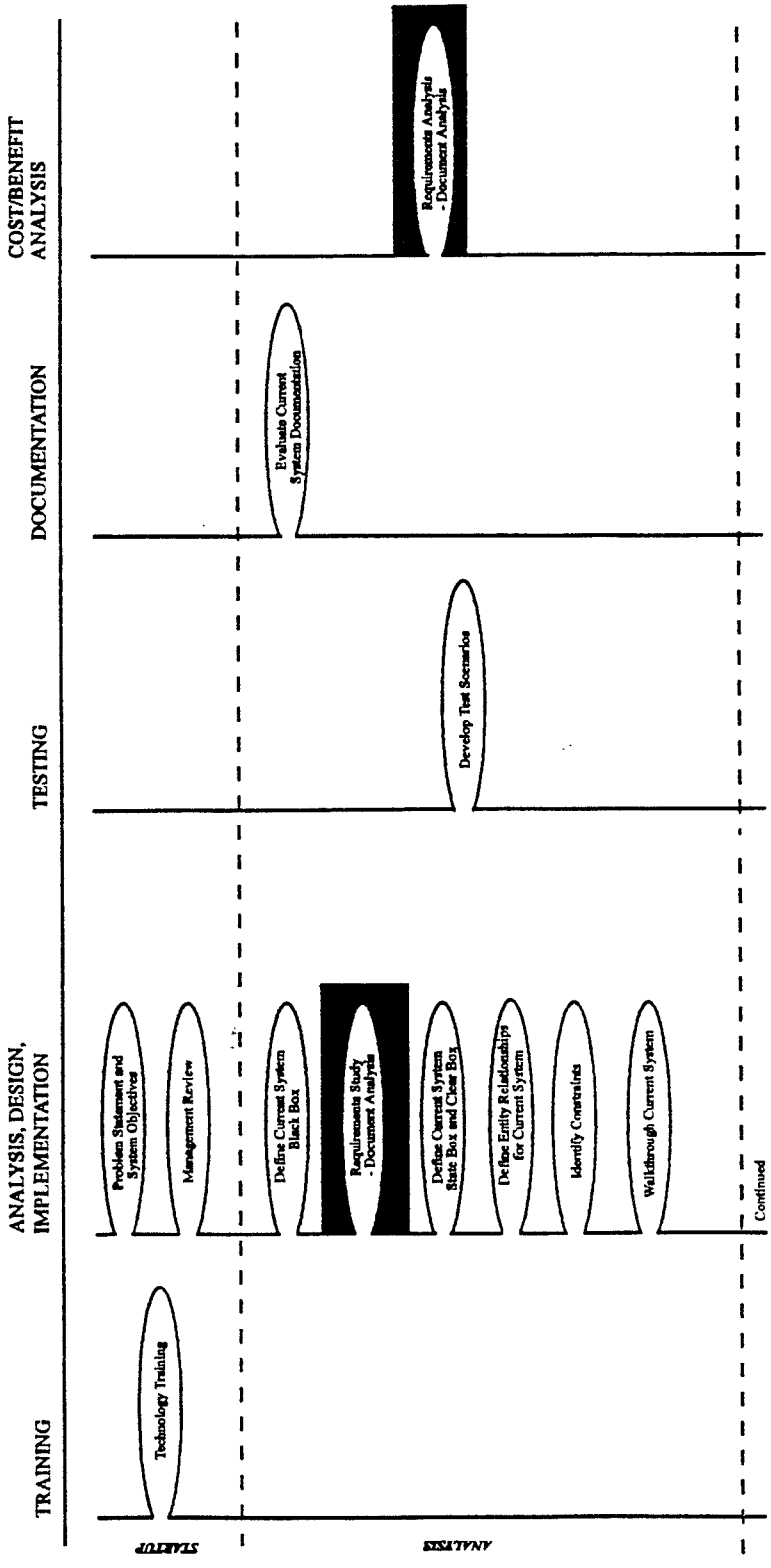


Figure 10. DIS Methodology - Document Analysis Activity

The paper was separated into documents, with a document being defined as:

- a specification sheet for one product
- a specification sheet for multiple products
- a marketing brochure describing a product family
- a white paper describing a product feature or functionality in detail (e.g., standards compliance, performance testing results)
- a published customer testimonial
- a press release describing a new product or products
- an annual report or financial profile of the vendor
- a price list for products or services
- a letter or memorandum

The stack of accumulated material was broken down into 250 documents, fifty of which were determined to be unsuitable for storage in the Tech Library. A few were disqualified because they represented duplicates of information previously examined. A large percentage of the rejects were cover letters which contributed no meaningful product information. The distribution of the remaining 200 documents is analyzed below. A tabulation by document is included as Appendix A.

As is shown in Figure 11, the largest volume of documents, 89 percent, are 8.5" x 11" in size. The 10 percent of the documents which are larger than letter size

may result in larger image files, depending on the content of the pages and the compression algorithms selected.

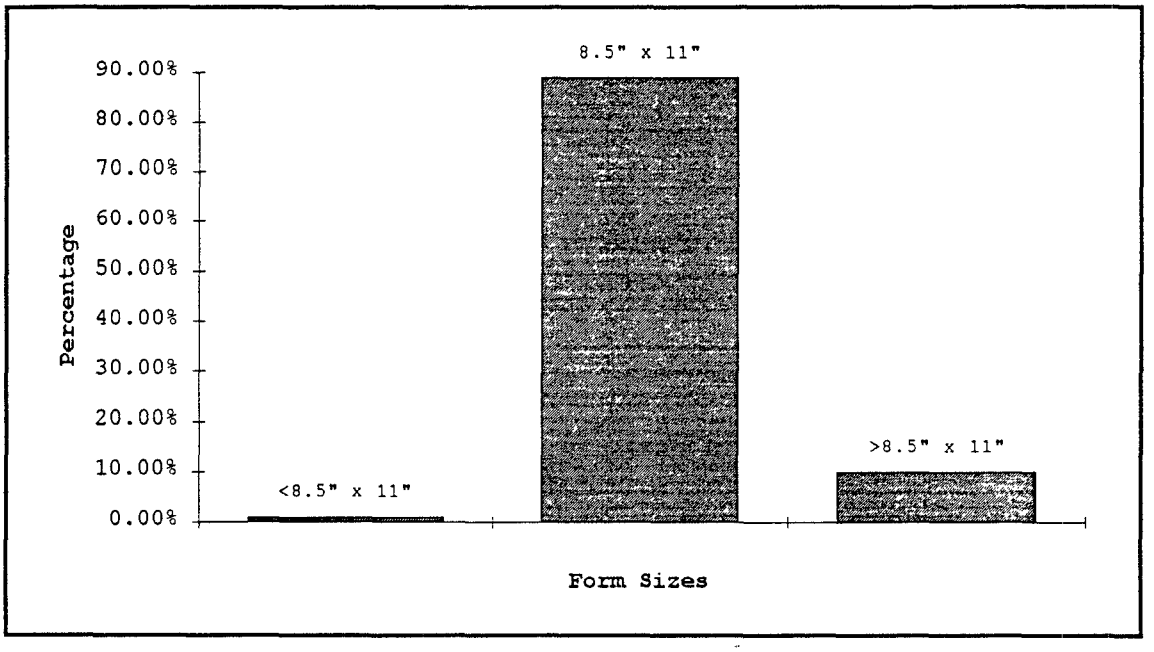


Figure 11. Distribution of Paper Sizes

Over 81 percent of the documents evaluated were double-sided. This will be a major consideration when selecting the type of scanner to be used for document capture.

Although 93.5 percent of the documents surveyed used black ink for text, the survey also found that a large number of vendors (68 percent) are now using colored ink on the documents they distribute. Even letters and press announcements were found to have at least one ink color in addition to black. This data is significant for evaluating the alternative image compression algorithms, as well as

the types of scanners, image displays, and printers to be configured.

Even though vendors have started using colored ink extensively, in this survey 86 percent are still printing their literature on white paper stock. Tinted or colored paper, used for 14 percent of the documents examined, can result in larger image files, depending on the scanner. Since colored stock is still being used for a relatively small percentage of documents, procedural methods, such as copying, may be the most feasible means for dealing with those documents prior to scanning.

The binding classification identifies not only how the pages of a document are fastened together, but also the number of images that would result from scanning. Figure 12 shows the three classifications: single sheets (single), a single sheet of paper that is folded once or twice (folded), and single sheets of paper that are folded once and bound together at the fold (bound). Interestingly, it shows that an approximately equal number of images would be generated by each classification, even though the majority of the documents are single sheets. There are an average of 3.7 images per document. These data, as well as the calculations for storage capacity, are input into the database-sizing activity during design.

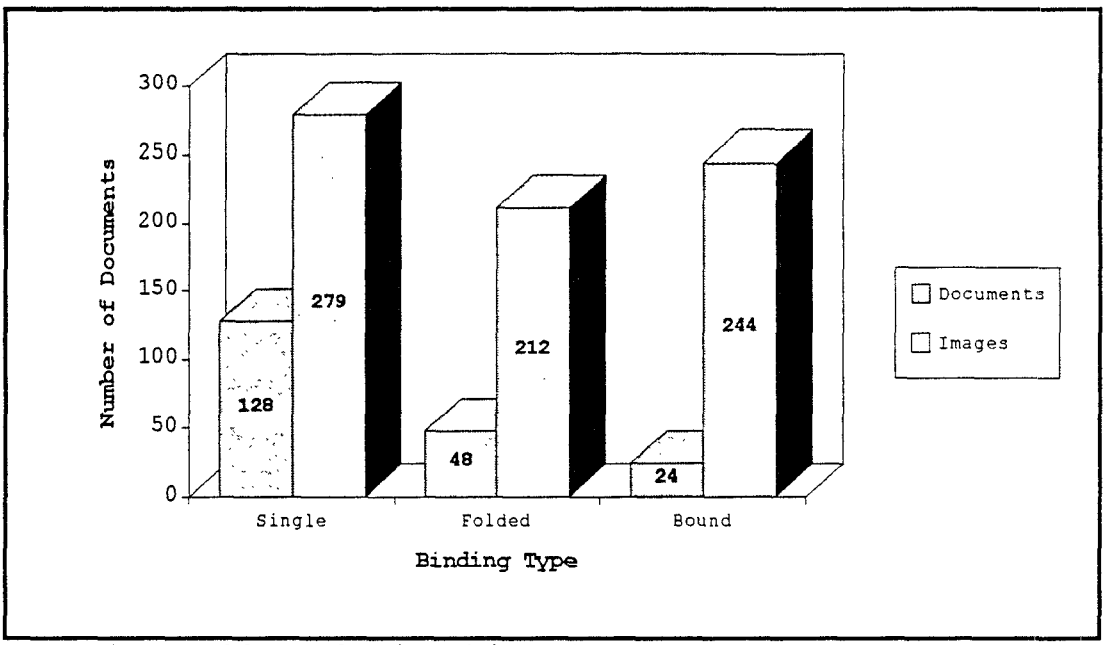


Figure 12. Relationship of Documents to Images

Of the 200 documents in the survey, 46 percent include some type of line art. Over 18 percent of the documents contain colored line art. Photographs are found in at least 47 percent of the documents and 26 percent of them contained color photographs. The use of photographs is largely found in hardware related documents, although some software companies are also beginning to include them.

Over 50 percent of the documents in the survey were in the software category, but the split between software and hardware documents is expected to be fairly even over time. The 22 percent of documents in the Co. (Company) Category consists of price lists, press releases, company background, company financial information, and other similar material.

The information from the Document Analysis activity will be used as input into a number of other activities. Within the Define Current System step, the classifications will be used during the evaluation of Image Compression Options. The information will also be used as input to the selection of hardware and software and will influence the design of the database and the user interface.

Reviewing the sample documents in Appendix B, the reader can see that the data provided for products varies greatly, even when the product category is identical. For instance, Cornerstone and Sigma Designs both have 120 dpi image display monitors, but the technical details about the products vary. Sigma Designs lists the phosphor as high efficiency, paper white, while Cornerstone's data sheet indicates that the phosphor type is P-104. As a result of these differences in attributes, generalized table elements have to be identified for the document index. This part of the document analysis is used as input to the prototype building activity, the definition of alternative system designs, and when building the data dictionary.

Image Compression Options

Figure 13 presents the activity loops in the methodology where the image compression analysis is conducted. The Requirements Study and Requirements Analysis loops are repeated for this work.

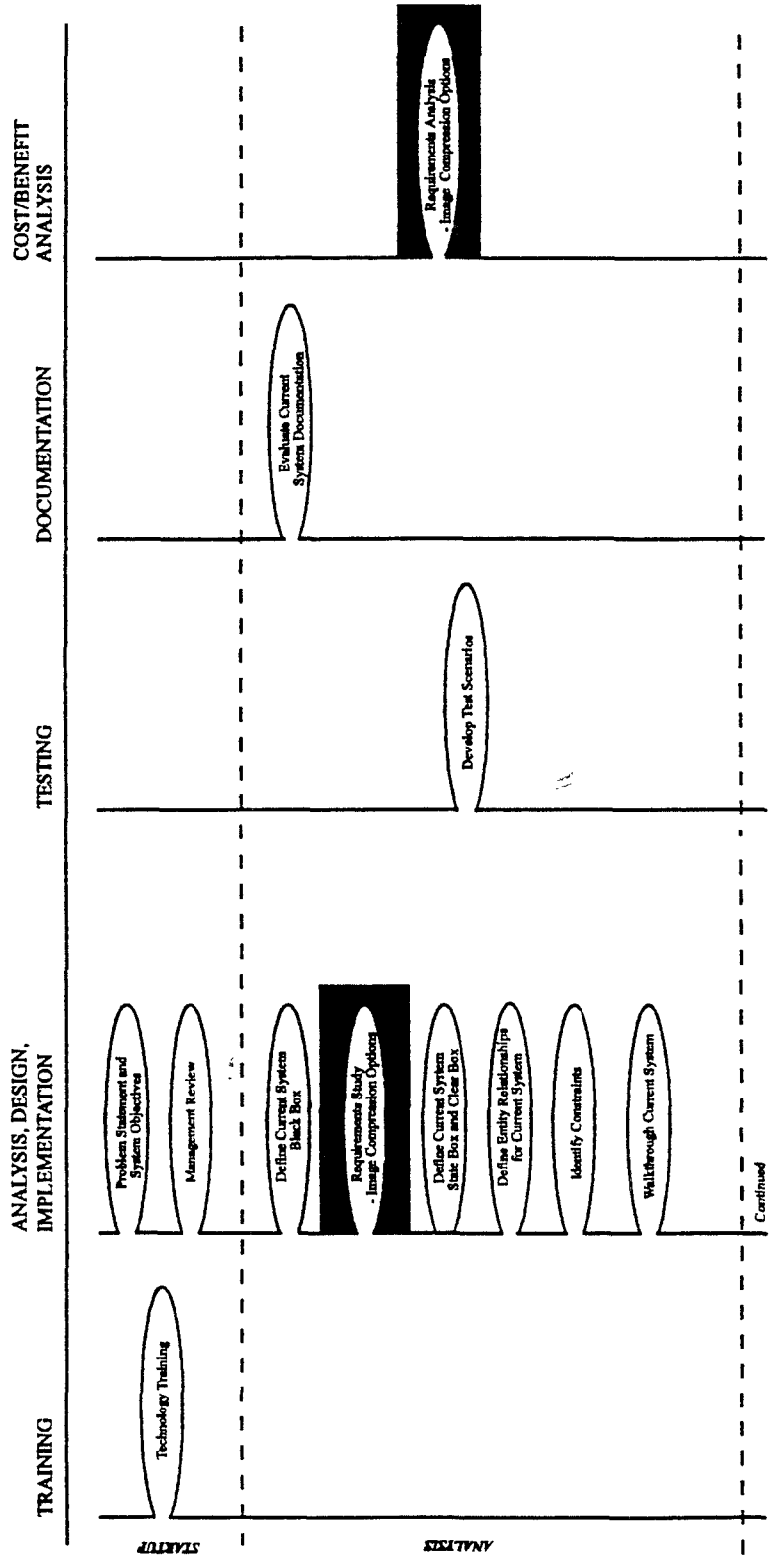


Figure 13. DIS Methodology - Image Compression Options

Document analysis established that many of the documents to be captured for the Tech Library contain color text and photographs. DISs that store color images are uncommon and very expensive at this time. The customer has set a constraint that a color solution cannot be proposed, so the only alternatives evaluated for storage of these images are bi-tonal and grayscale representations. Grayscale provides better image resolution, but also requires a more expensive system, including greater disk capacity and compression/decompression accelerators to support the JPEG algorithm. CCITT Group 4, typically used for bi-tonal images in DISs, provides a less costly system but with some sacrifice in display quality of images, especially those with photographs.

Figure 14 contains estimates of the file sizes for 8.5" x 11" documents at various compression ratios using both CCITT Group 4 (bi-tonal) and JPEG (grayscale). For JPEG, the Lossy option produces the best compression factor, but with a greater loss in image quality. The Exact option for the algorithm ensures that no pixel (bit) of image data is lost, but at a dramatic increase in compressed file size.

Bi-tonal images

	Raw Image Size	Best Case CCITT Group 4		Worst Case CCITT Group 4		Expected CCITT Group 4	
		Ratio	Size	Ratio	Size	Ratio	Size
200 dpi	467KB	28:1	16.7KB	7:1	66.8KB	15:1	31.2KB
300 dpi	1.05MB	28:1	37.5KB	7:1	150.2KB	15:1	70.1KB

8-bit Grayscale images

	Raw Image Size	Best Case JPEG (Lossy)		Worst Case JPEG (Exact)		Expected JPEG	
		Ratio	Size	Ratio	Size	Ratio	Size
200 dpi	3.74MB	30:1	125KB	2:1	1.87MB	20:1	187KB
300 dpi	8.42MB	30:1	281KB	2:1	4.21MB	20:1	401KB

Figure 14. Comparison of Compression Ratios and File Sizes

The expected annual volume for the Tech Library is 7,990 documents. If, on average, there are 3.7 image pages per document, there will be 29,563 image files to be stored on the system annually. Using the compression ratios in the expected range shown in Figure 14, the required optical capacities are presented in Figure 15. The 650MB platter used in the example is a common 5.25" optical disk capacity.

Bi-tonal Images

	Expected File Size	Capacity Required	# 650MB Platters Required
200 dpi	31.2KB	922MB	1.42
300 dpi	70.1KB	2.1GB	3.19

8-bit Grayscale Images

	Expected File Size	Capacity Required	# 650MB Platters Required
200 dpi	187KB	5.5GB	8.51
300 dpi	401KB	11.9GB	18.24

Figure 15. Optical Disk Capacity Estimates

Since a large number of images contain photographs and complex graphics, it is unlikely that the expected ratios could be achieved consistently using CCITT Group 4 compression. Consequently, the calculations in Figure 16 represent 50 percent of the documents at the Expected ratio and 50 percent using the Worst Case ratio to adjust the capacity. Even using this adjustment, the Group 4 option requires less optical disk capacity than JPEG.

Bi-tonal Images with Adjustment

	Expected File Size	Capacity Required	Worst Case File Size	Capacity Required	# 650MB Platters Required
200 dpi	31.2KB	922MB	66.8KB	987MB	2.22
300 dpi	70.1KB	2,072MB	150.2KB	2.2GB	5.0

Figure 16. Bi-tonal Capacity using Photographs

Based on the tables in Figures 14, 15, and 16 above, it is obvious that the use of grayscale images will require a larger investment in optical disk media than bi-tonal images. Additional factors that have an adverse impact on the system cost include: the use of an optical jukebox to facilitate access to optical disks; more complex image compression and decompression software requiring expensive hardware accelerators; and more costly image display monitors and controllers.

While the difference in the initial cost of the server-based components for the two alternatives only varies by the cost of the optical media, the cost of storing grayscale images quickly outpaces bi-tonal images in the following years. The capacity of the jukebox used in Figure 17 is sufficient for only one year of the 200 dpi, JPEG-compressed, grayscale images, whereas is it can support four years of bi-tonal image storage using the adjusted image compression ratios in Figure 16. The greater than 50 percent delta in workstation costs is also significant.

	CCITT G 4	JPEG
Server-based Component Prices		
Hewlett-Packard Jukebox (C1718C)	$6,700	$6,700
HP 650MB Rewritable Optical Disks (8 pack)	$1,360	$2,720
Total Server-based Prices	$8,060	$9,420
Workstation Component Prices		
Optibase Model 100 JPEG Accelerator Board		$695
Optibase Workshop JPEG Software		$149
Cornerstone DualPage 120 Monitor w/Polished Glass	$1,495	$1,495
Cornerstone DualPage 120, Monochrome Controller	$1,000	
Cornerstone DualPage 120, Grayscale Controller		$1,500
Per Workstation Cost	$2,495	$3,839
Incremental cost per workstation for JPEG		$1,344

Figure 17. Price Comparison Between Alternatives

Additional factors which impact the cost of implementing the grayscale option include the higher prices of scanners and laser printers that support grayscale. Perhaps even more significant is the cost of integrating the grayscale support into the image management software that has been selected for use in implementing the Tech Library. The ViewStar image engine provides native support for bi-tonal images and CCITT Group 4 compression. The support for grayscale images would have to be designed and developed as a separate subsystem. This would require significant engineering resources, which are not available for this project. The figures presented in Figure 17 represent list prices for products required by the alternative solutions

The bi-tonal image alternative is the preferred option for the Tech Library implementation, for both cost and technical support reasons. The goal for the system is to provide access to the specifications of a product. Accurate reproduction of the photographs is not required to support this goal. As a result, CCITT Group 4 compression of bi-tonal images is considered to be the best option for implementing the Tech Library.

Output of this analysis is used as input during the selection of system components and alternative system designs.

Prototype Development

Input:	Approved Problem Statement Approved System Objectives Document Analysis Current Logical System Description Black Box Descriptions for Alternatives
Output:	Working Prototype

During this activity, a working prototype system is created. The prototype may not be a fully functional system, but should provide the model for the user interface. This model can be used in parallel activities for user training and system testing. The work conducted for this thesis relative to prototype development is presented below.

As the analysis of documents progresses, the prototype user interface is refined. This is an iterative process, whereby users operate the test system during training can contribute comments about its strengths and weaknesses. These comments are input to refining the prototype and to other system development activities. Prototype testing and training may identify constraints which are then input into the Identify Constraints step. These testing and training activities also become adjuncts to the activities of defining system alternatives and designing the new system. Since the framework for the methodology provides the ability to go back and repeat earlier activities based on new discoveries, the Develop Black Box Alternatives activity can be repeated to incorporate the results of the prototyping activity.

Because the DIS alters the way the users do their work so significantly, the interaction of the user with the prototype is essential to getting the user interface design right prior to deployment. The use of the protype in the Testing activities helps to ensure that the system developers are not straying away from the stated requirements for the DIS.

User Interface Design

The Prototype Development loop is highlighted in Figure 18. Additionally, Prototype Training, Execute Prototype Tests, and Evaluate Prototype activities are highlighted to indicate the parallel activities in the Training, Testing and Cost/Benefit Analysis spirals.

The Document Analysis activity, which is conducted during the first iteration of the Requirements Study loop, provides much of the input to the initial prototype design. While studying the documents, the context is examined to determine what information is important. As the system developer determines alternative implementations for the DIS, the alternatives are used as input to the prototype building process. In some cases, more than one prototype may be built. With a preliminary structure for the database tables, an early prototype can be built.

The test application for this case study allows users to experiment with the Tech Library before it is completed. The results of this testing are fed back into the prototyping activity and are also output to database design.

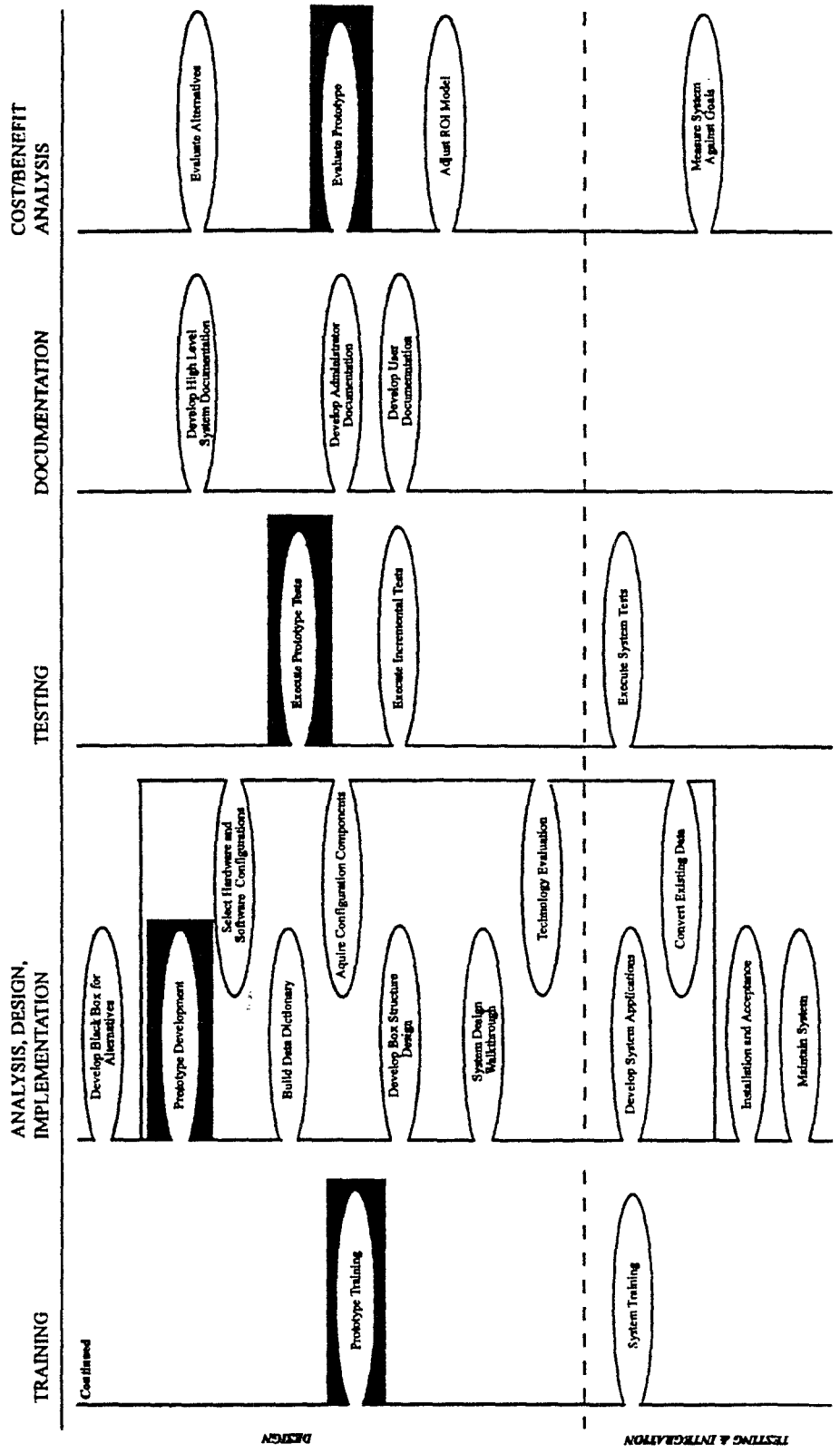


Figure 18. DIS Methodology - User Interface Design

While this appears to mirror the work done in developing a typical data processing application's user interface, the developers must use their knowledge of DISs to determine how to best support the user's needs. Since much of the important information on the document is not in the database, and thus not directly available for querying, the developer must be cognizant of the different ways the user may want to access the documents. Keying all of the data from the document into the database is not generally feasible due to time and space constraints. So the user interface must provide a generalized notion of the data, allowing the user to identify candidate documents that can be reviewed in detail for more specific information.

As the developer approaches the design and implementation of a user interface as part of the prototype, it is important to consider that available image management software provides a wide range of functionality and capabilities. Some software is designed as a toolkit, allowing the developer great flexibility in creating a custom system. For this type of software, the development effort will most likely be more time consuming and complex. Other image management software allows a system to be built easily and quickly, but imposes certain constraints on the developer's creativity. As a result, the user interface may have to be designed to accommodate specific limitations of the software.

The tool used to develop the prototype user interface for the Tech Library is REDAC, a product of Severn Companies, Inc. REDAC is built on top of the ViewStar image management software package. ViewStar provides the image management engine; REDAC provides a customizable interface to the engine. The time and cost of developing a system are greatly reduced by having a non-programmatic application builder, but, as noted above, some flexibility in user interface design is sacrificed.

For this project, the prototype is used to: 1) test the structure of the screens to ensure the user interface is easy to use and 2) validate that data elements to be stored in the database support queries which retrieve relevant documents. By selecting a tool that provides an easily generated user interface, these tests can be conducted early in the design process and thus have a positive effect on the database design and implementation.

The data elements for the Tech Library tables are listed in Appendix C. The elements represent relevant information which is generally present on technical literature. The first prototype screens were developed from that list. Figure 19 is the screen used to define the template for the Library TECHDOC. Figure 20 is the template for the VENDORS table.

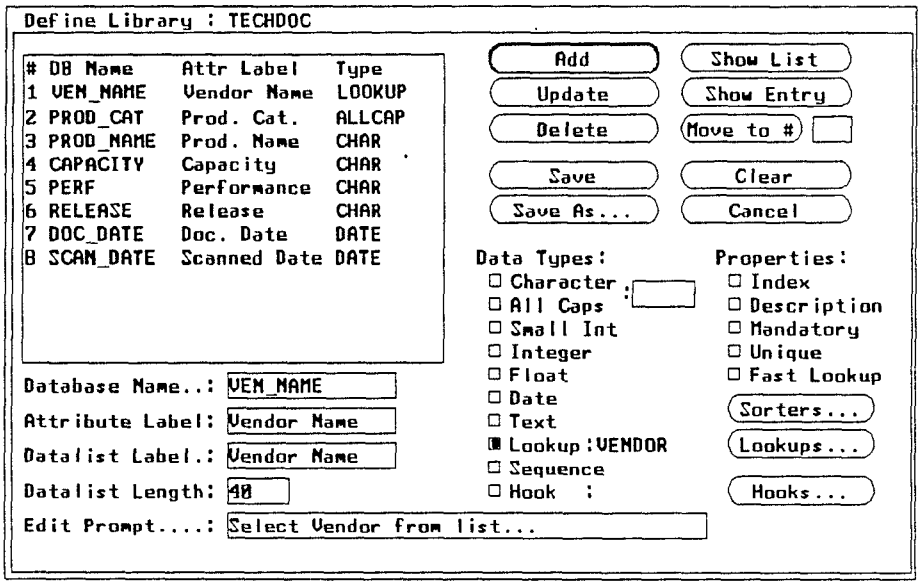


Figure 19. Template for Initial Library Definition

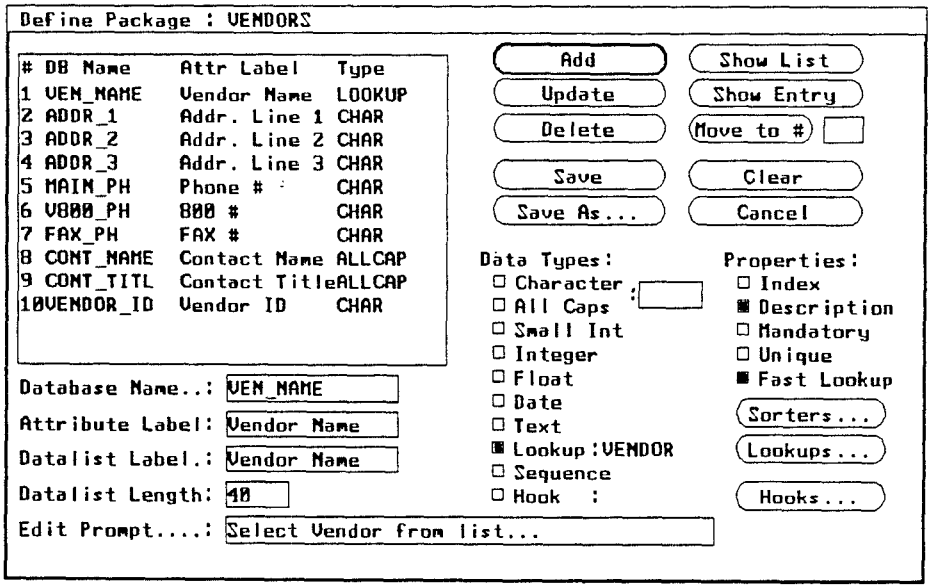


Figure 20. Template for Initial Package Definition

Several engineers were asked to use the prototype system to capture technical documents and associated data. They also attempted to query the prototype database to find specific products. On the basis of their comments, the database and screens were modified to include additional data elements and to rearrange their presentation on the data entry and data list screens.

The modified tables, VENDORSR and TECHDOCR, are presented in Figures 21 and 22. The majority of the changes have to do with the relative importance of the data in each element. For example, it is generally more important to have a vendor's telephone number visible on a data list screen than to have the vendor's address. Therefore, the main phone number data element was moved next to the vendor name so that they display close together. This reduces the need for the user to search the screen for a data element that is frequently used. Similarly, it is more important to see the product category and name, than to see the vendor name, when initially searching for products. Consequently, the vendor name was moved down in the TECHDOCR table, allowing the product category, product name, and capacity to be displayed simultaneously when a product list is displayed (see Figure 23). Figure 24 is the revised data entry screen for the documents.

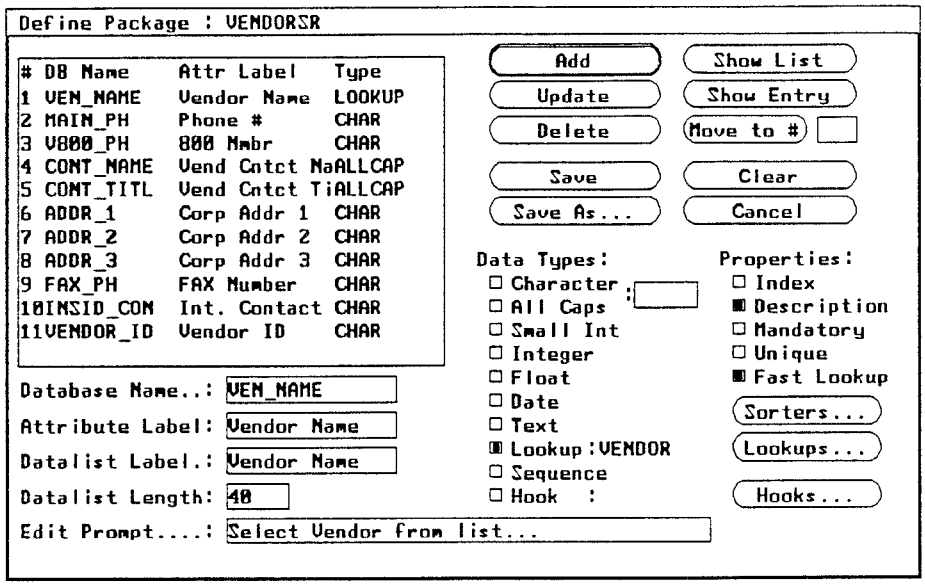


Figure 21. Revised VENDORS Template

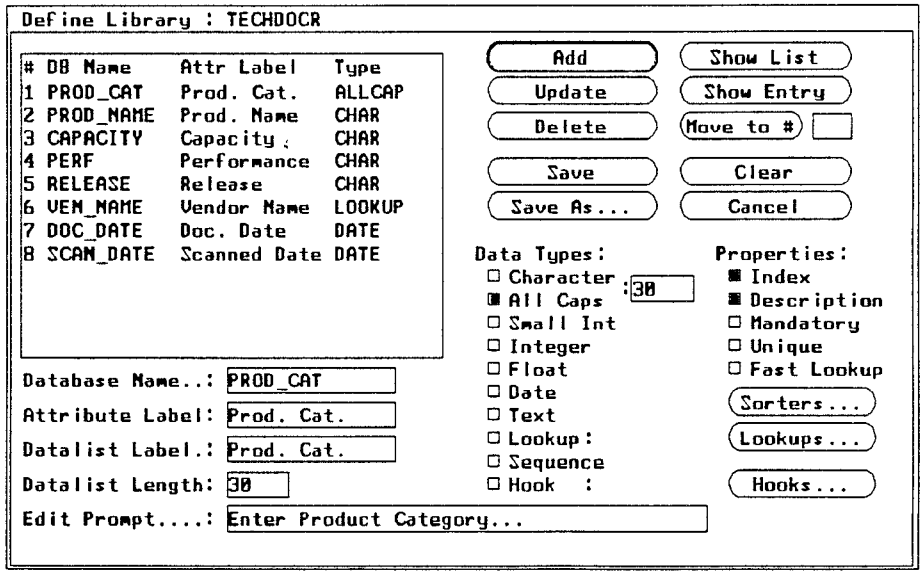


Figure 22. Revised TECHDOC Template

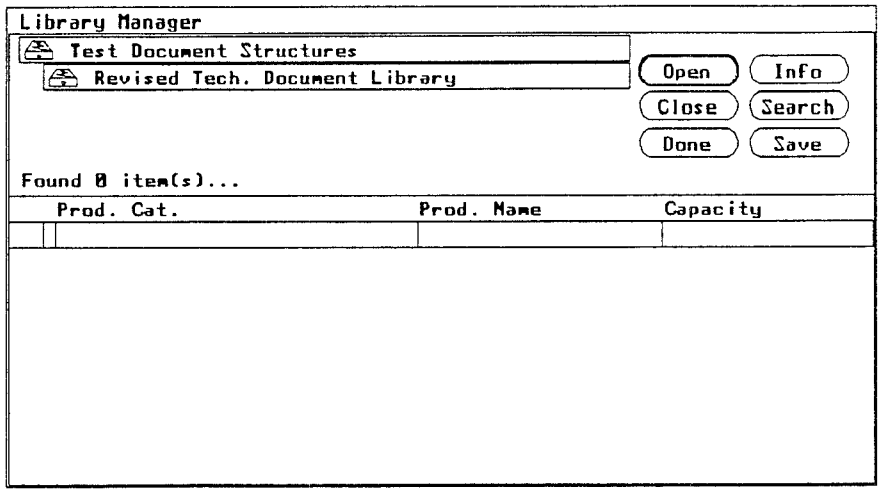


Figure 23. Revised TECHDOC Data List Screen

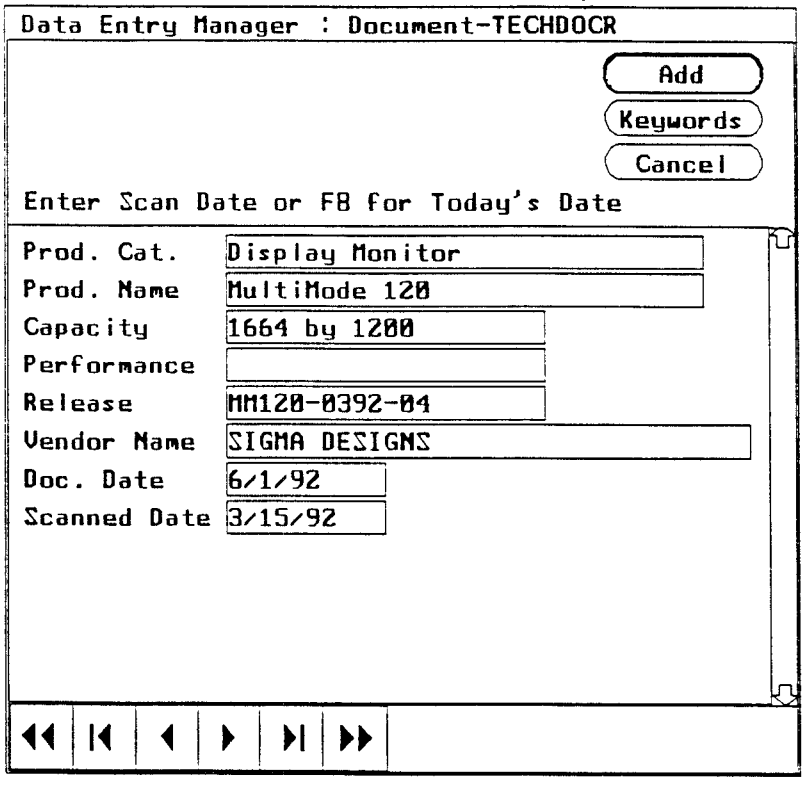


Figure 24. Revised TECHDOC Data Entry Screen

The revised structures for both tables are available in Appendix D.

As mentioned earlier, the design of the user interface takes input from the Document Analysis activity to identify what constitutes a document and how it is best referenced. The results of the prototype testing flow directly into the final database and user interface design.

Summary

Many more tasks must be carried out to implement the complete Tech Library system. The three activities presented in this chapter provide a portion of the foundation for the remainder of the work to be completed. The activities performed for this case study demonstrate that the proposed DIS methodology is viable. It is flexible enough to accommodate unique and complex DIS attributes. In addition, the methodology supports the recursive nature of many of the activities, such as prototyping. Output from one activity flows into one or many others as work progresses, even allowing information to be shared between activities in parallel spirals.

Three DIS-specific activities are performed in this case study:

- Document Analysis -- Documents to be stored in the system are analyzed to determine the features that are relevant to document capture, storage, and retrieval.

- Image Capacity Analysis -- Sizing of the image storage capacity is performed using data about the types of documents to be stored, the volumes, and the available image storage and compression options.
- User Interface Design -- Users are involved in testing the prototype user interface, allowing developers to adjust the design to accommodate user requirements.

The balance of the system development effort is performed in the same manner using the proposed DIS methodology. Based on the experience with the three activities included in the case study, the methodology should be effective for the remainder of the activities.

CHAPTER 7. CONCLUSIONS AND RECOMMENDATIONS

While DISs provide the technology needed to store the electronic equivalent of a piece of paper, it is unlikely these systems will totally replace paper until developers deliver systems that closely fit the needs of the organization and its staff. The system developer has a handicap in attempting to do this, however. DISs introduce requirements which are not critical for other information systems and the traditional systems development methodologies do not do an adequate job supporting the DIS developer. Consequently, the system developer may resort to using a hodgepodge of methodologies or no methodology at all. The proposed DIS methodology in Chapter 5 provides a starting point for addressing this problem. The case study in Chapter 6 begins the evaluation process for the DIS methodology.

Lessons Learned

One of the most important lessons learned from this thesis project is that it is considerably easier to know that something is needed and to critique the way things are currently done than it is to define the solution. Many papers in the trade literature point out the problems without offering comprehensive solutions. Perhaps the complexity of the task should have been intuitively obvious

based on the number of books and articles written on the topic of methodologies. In actuality, the task of developing a methodology was, at times, quite overwhelming. This thesis provides a framework for additional research and enhancement of the proposed DIS methodology, anticipating that it can eventually be considered for general use.

Benefits and Drawbacks

As it is defined in Chapter 5, the DIS methodology does provide the flexibility necessary for repeating activities that cannot be finalized by doing them once. This is especially important since it is difficult for developers and users to see the full potential for a DIS prior to its implementation.

The developer also have the ability to make adjustments in activities, by adding, deleting, or moving them up or back in the schedule. This can be done without breaking or subverting the methodology.

Another strength in the proposed DIS methodology is that parallel activity spirals can be started. This is well-suited to the way a DIS is typically implemented.

The proposed DIS methodology also allows the system developer to draw upon the knowledge of the users and incorporate their input into the system development process. The human factors element is critical because

DISs so dramatically change the way the users do their jobs.

The methodology allows the document imaging technology and a prototype system to be introduced to the user very early in the life cycle via the Technology Training and Prototype Training activities. By providing this type of support, the proposed DIS methodology should help to eliminate the uneasiness expressed in the following quote from an Air Force decisionmaker (Boehm 1973).

> "You software guys are too much like the weavers in the story about the Emperor and his new clothes. When I go out to check on a software development the answers I get sound like, "We're fantastically busy weaving this magic cloth. Just wait a while and it'll 'look terrific'." But there's nothing I can see or touch, no numbers I can relate to, no way to pick up signals that things aren't really all that great. And there are too many people I know who have come out at the end wearing a bunch of expensive rags or nothing at all."

Because a DIS has such a significant impact on the organization and the users, the participation of the user is much more important than for traditional data processing systems. The proposed methodology provides opportunities for users to be involved as they define the way they currently handle assigned tasks and what would help them do their job better.

A critical look at the proposed methodology shows that there is at least one aspect that should be addressed more

fully. There are few methods and tools recommended for the developer to use in carrying out each of the activities. System developers would undoubtedly benefit from some guidance in this area.

Additionally, it is unclear if this type of methodology will be well received by organizations who bid on fixed price contracts. The task of estimating work to be done at a fixed price is formidable for most system developers. There are so many variables that cannot be seen up front. Organizations that have historical data available from previous projects have a distinct advantage. However, with a flexible methodology, such as the one proposed in this thesis, the task is likely to be seen as impossible.

Future Research

One area where additional research should be conducted involves the types of methods, tools, and documentation that are appropriate at various stages. The options for diagrams, specifications, and other supporting documents are almost completely wide open in the proposed DIS methodology. However, to make it more practical rather than theoretical, most system developers would welcome suggestions of what could be used at various stages.

As a next step in researching this methodology, the system development process should be followed through to

the end using the proposed methodology. This research should also include practical suggestions for the supporting methods, tools, and documentation that are useful during each activity.

The case study of the development of the technical documentation library was well supported by the proposed methodology. It was, however, a small system when compared to many DISs being developed by corporations, state governments, and the federal government. To establish that the methodology is scalable, it should be used to develop a large DIS. This, of course, cannot be done until it has been completely validated on smaller systems.

Summary

After further analysis, it may be found that another methodology could be adapted to fit many of the needs of the DIS developer. The proposed DIS methodology, however, provides a very adaptable approach to systems development that is not apparent with the other methodologies examined in Chapter 2. As a result, it offers a dynamic environment to support the idiosyncrasies of a DIS.

APPENDIX A - Document Analysis Data

RESULTS OF DOCUMENT ANALYSIS

	Form Size			Sided		Ink		Paper		Binding			Graphics				Category		
	<8.5x11	8.5x11	>8.5x11	Single	Double	Black	Colored	White	Colored	Single	Folded	Bound	Line Art	Colored Line Art	Photo	Colored Photo	S/W	H/W	Co.
1		X		X		X		X		1									X
2		X		X		X		X		1									X
3		X		X		X		X		4									X
4		X			X	X	X	X				8	X				X		
5		X			X	X		X		1							X		
6		X			X	X		X		1			X				X		
7		X			X	X		X			3		X				X		
8		X			X	X		X		2			X				X		
9		X			X	X		X				16							X
10		X			X	X		X				7	X				X		
11		X			X	X		X		2			X				X		
12		X			X	X		X			5		X				X		
13		X			X	X		X				12	X		X		X		
14		X			X	X		X			4		X		X		X		
15		X			X	X		X			3				X		X		
16		X			X	X		X			4				X		X		
17		X			X	X		X		2			X				X		
18		X			X	X		X		6							X		
19		X			X	X	X	X			5		X	X			X		
20		X			X	X		X		4									X
21		X			X	X		X		2							X		
22		X			X	X		X		2			X				X		
23		X			X	X		X			4		X				X		
24		X			X	X		X		2			X				X		
25		X			X	X		X		2			X				X		
26		X			X	X		X			4		X				X		
27		X			X	X		X			4		X				X		
28		X			X	X		X			4		X				X		
29		X			X	X		X		2			X				X		
30		X			X	X		X		2			X				X		
31		X			X	X		X		2			X		X		X		
32		X			X	X	X	X				12	X						X
33		X			X	X	X	X			5		X		X		X		
34		X			X	X		X				21	X						X
35		X		X		X		X		15			X						X
36		X		X		X		X		7			X				X		
37		X		X		X		X		6								X	X

RESULTS OF DOCUMENT ANALYSIS

	Form Size			Sided		Ink		Paper		Binding			Graphics				Category		
	<8.5x11	8.5x11	>8.5x11	Single	Double	Black	Colored	White	Colored	Single	Folded	Bound	Line Art	Colored Line Art	Photo	Colored Photo	S/W	H/W	Co.
38		X			X	X	X	X				14	X	X			X		
39		X			X	X		X		13			X				X		
40		X			X	X	X	X			4		X		X	X	X		
41		X			X	X	X	X			4				X		X		
42			X		X	X	X	X		2			X					X	
43			X	X		X	X	X		1									X
44		X		X		X	X	X		1							X		
45		X			X	X	X	X		2							X		
46		X		X		X	X	X		1							X		
47		X			X	X	X	X		2							X		
48		X			X	X	X	X		2			X				X		
49		X			X	X	X	X		2							X		
50		X			X	X	X	X		2			X				X		
51		X		X		X	X	X		1									X
52		X			X	X	X	X		2							X		
53		X		X		X	X	X		1							X		
54		X		X		X	X	X		1							X		
55		X		X		X	X	X		1							X		
56		X			X	X	X	X		2									X
57		X		X		X	X	X		1									X
58		X		X		X	X	X		1							X		
59		X			X	X		X				9	X						X
60		X		X			X	X		1							X		
61		X			X	X	X		X	2					X		X		
62		X			X		X		X	2									X
63		X		X			X		X	1									X
64		X		X			X		X	1							X		
65		X		X			X		X	1							X		
66		X		X			X		X	1									X
67		X		X			X		X	1									X
68		X		X			X		X	1									X
69		X			X	X	X	X		4			X	X	X	X			X
70		X			X	X	X	X		4			X	X	X				X
71		X		X			X	X		1							X		
72		X		X			X	X		1							X		

RESULTS OF DOCUMENT ANALYSIS

	Form Size			Sided		Ink		Paper		Binding			Graphics				Category		
	<8.5x11	8.5x11	>8.5x11	Single	Double	Black	Colored	White	Colored	Single	Folded	Bound	Line Art	Colored Line Art	Photo	Colored Photo	S/W	H/W	Co.
73		X			X	X	X		X	2			X	X			X		
74		X			X	X	X	X		2					X	X		X	
75		X			X	X	X	X				10	X	X	X	X		X	
76		X			X	X	X	X		2			X	X			X		
77		X		X		X		X		2									X
78		X			X	X		X		2									X
79		X			X	X	X	X			4		X	X	X	X		X	
80		X			X	X	X	X			4		X	X	X	X		X	
81		X			X	X	X	X			4		X	X	X	X		X	
82		X			X	X	X	X			4		X	X	X	X		X	
83		X		X		X		X		2									X
84		X			X	X	X	X		2					X	X	X		
85		X		X		X		X				20			X		X	X	X
86		X		X		X		X		1								X	
87		X			X	X		X			4				X			X	
88			X		X	X	X		X			8	X	X	X	X	X	X	
89		X		X		X		X		2			X					X	
90		X			X	X	X	X				8	X	X	X	X	X	X	
91		X			X	X	X	X		2			X		X			X	
92		X			X	X	X	X			4		X		X			X	
93		X			X	X	X	X		2					X			X	
94		X			X	X	X	X			4		X		X			X	
95		X			X	X	X	X		2					X			X	
96		X			X	X	X	X			4		X		X			X	
97		X			X	X	X	X			4		X		X			X	
98		X			X	X	X	X		2					X			X	
99		X			X	X	X	X		2			X		X			X	
100		X			X	X	X		X		4				X	X	X	X	
101		X			X	X	X	X		2					X	X		X	
102		X			X	X	X		X		4				X	X		X	
103		X			X	X	X	X		2					X			X	
104		X			X	X	X	X		2					X	X		X	
105		X			X	X	X	X		2					X	X		X	
106		X			X	X	X	X		2					X	X		X	
107		X			X	X	X	X		2					X	X		X	
108		X			X	X	X	X				11	X		X	X	X	X	
109		X			X	X	X	X				7	X		X	X	X	X	

RESULTS OF DOCUMENT ANALYSIS

	Form Size			Sided		Ink		Paper		Binding			Graphics				Category		
	<8.5x11	8.5x11	>8.5x11	Single	Double	Black	Colored	White	Colored	Single	Folded	Bound	Line Art	Colored Line Art	Photo	Colored Photo	S/W	H/W	Co.
110		X			X	X	X	X				7	X	X			X		
111		X			X	X	X	X				8	X		X	X		X	
112		X			X	X	X	X			4		X		X	X		X	
113		X			X	X	X		X			8	X		X	X		X	
114			X		X	X	X	X				4	X		X	X	X		X
115		X			X	X	X	X			4		X	X			X		
116		X			X	X	X	X			4				X				X
117		X			X	X	X	X			6		X		X	X	X		X
118		X			X	X	X	X		2					X	X		X	
119		X			X	X		X		1					X	X		X	
120		X			X	X		X		1							X		
121		X			X	X	X	X		2			X				X		
122		X			X	X	X	X		2					X	X		X	
123		X			X	X	X	X		2					X	X		X	
124		X			X	X		X		4							X		X
125		X		X		X		X		1							X		
126		X		X		X		X		1							X		
127		X			X		X	X			6		X	X			X		
128		X			X		X	X			6		X	X			X		
129		X			X		X	X			6		X	X	X	X	X		
130			X		X	X	X	X			12		X	X	X	X	X	X	X
131		X			X	X	X	X		2					X	X	X		
132		X			X	X	X	X		2			X		X	X	X		
133		X			X	X	X	X		2			X		X	X	X		
134		X			X	X	X	X		2			X		X	X	X		
135		X			X	X	X	X		2			X		X	X	X		
136		X			X	X	X	X		2			X		X	X	X		
137			X		X	X	X		X		4		X	X					X
138		X			X	X	X	X		2			X	X			X		
139		X			X	X		X			3		X				X		
140			X		X	X	X		X	2					X			X	
141			X		X	X	X		X	2					X			X	
142			X		X	X	X		X	2					X			X	
143			X		X	X	X		X	2					X			X	
144			X		X	X	X		X	2					X			X	
145			X		X	X	X		X	2					X			X	
146			X		X	X	X		X	2					X			X	

RESULTS OF DOCUMENT ANALYSIS

	Form Size			Sided		Ink		Paper		Binding			Graphics				Category		
	<8.5x11	8.5x11	>8.5x11	Single	Double	Black	Colored	White	Colored	Single	Folded	Bound	Line Art	Colored Line Art	Photo	Colored Photo	S/W	H/W	Co.
147			X		X	X	X		X	2					X			X	
148		X			X	X		X			4		X		X			X	
149			X	X		X	X		X	1					X			X	
150			X		X	X	X		X	2							X		
151			X		X	X	X		X	2								X	
152			X		X	X	X		X	2					X			X	
153			X		X	X	X		X	2					X			X	
154		X			X	X		X			6							X	
155		X			X	X	X	X				11			X				X
156		X			X	X		X		4					X		X	X	
157		X			X	X		X				8			X				X
158		X			X	X		X		2							X		
159		X		X		X		X		1					X				X
160		X			X	X		X		2			X						X
161			X		X	X	X		X		4		X	X	X	X	X		
162	X				X	X	X	X			4				X	X		X	
163		X		X		X		X		1					X			X	
164		X			X	X	X	X		2								X	
165		X			X	X	X	X		2					X	X		X	
166		X			X	X	X	X		2					X	X		X	
167		X		X		X		X		1									X
168		X			X	X	X	X		2									X
169		X			X	X		X		6									X
170		X			X	X	X	X			4		X	X	X	X	X	X	
171		X			X	X	X	X		2					X	X		X	
172		X			X	X	X	X		2					X	X		X	
173		X			X	X		X		2					X		X	X	
174		X			X	X		X		3									X
175		X			X	X		X		4									X
176		X			X	X	X	X				12	X	X	X	X	X	X	
177		X			X	X	X	X				7	X	X	X	X			X
178		X			X	X	X	X				8	X	X	X	X		X	
179		X			X	X	X	X				8	X	X	X	X		X	
180		X			X	X	X	X			2				X	X	X		
181		X			X	X		X		2			X				X	X	
182		X			X	X		X		2					X			X	
183		X			X	X		X		2							X	X	
184		X			X	X	X	X			6		X	X	X	X	X		

RESULTS OF DOCUMENT ANALYSIS

	Form Size			Sided		Ink		Paper		Binding			Graphics				Category		
	<8.5x11	8.5x11	>8.5x11	Single	Double	Black	Colored	White	Colored	Single	Folded	Bound	Line Art	Colored Line Art	Photo	Colored Photo	S/W	H/W	Co.
185		X			X	X	X	X		2							X		
186		X		X		X	X	X		1							X		
187		X			X	X		X		2									X
188		X			X	X		X		2									X
189		X			X	X	X	X		2			X	X					X
190		X			X	X	X	X		2			X	X					X
191		X			X	X	X	X			4		X	X			X		
192		X			X	X	X	X			4		X	X			X		
193		X			X	X	X	X			4		X	X			X		
194		X		X		X		X		1							X		
195		X			X	X	X	X			4		X	X			X		
196		X			X	X	X	X			4		X	X			X		
197		X			X	X	X	X			4		X	X	X	X	X		
198		X			X	X	X	X		2					X	X	X	X	
199		X			X	X	X	X		2					X	X	X	X	
200	X				X	X	X	X			6		X				X		
										279	212	244							

SUMMARY RESULTS FOR DOCUMENT ANALYSIS

Form Size			Sided		Ink		Paper		Binding			Graphics				Category		
<8.5x11	8.5x11	>8.5x11	Single	Double	Black	Colored	White	Colored	Single	Folded	Bound	Line Art	Colored Line Art	Photo	Colored Photo	S/W	H/W	Co.
Total Column Counts																		
2	178	20	37	163	187	133	172	28	128	48	24	92	37	94	53	103	73	45
Total Documents		200																
Percent of Total																		
1.00%	89.00%	10.00%	18.50%	81.50%	93.50%	66.50%	86.00%	14.00%	64.00%	24.00%	12.00%	46.00%	18.50%	47.00%	26.50%	51.50%	36.50%	22.50%
									Total Image Count									
									279	212	244							
									Percent of Total Image									
									37.96%	28.84%	33.20%							

DualPage™ 120

The DualPage 120 comes to you as a complete system—with video controller card, display, video cable, software drivers, documentation technical support and one-year warranty.

Software Supported

OS/2 and Presentation Manager 1.2 and 1.3 (IBM; AT, PS/2)

Windows 3.0 and 3.1 (Microsoft)

Windows applications such as: Pagemaker (Aldus®), Excel (Microsoft), Amu (Samna), Ventura (Ventura)

Ventura Publisher (Ventura)

AutoCAD, AutoSketch (Autodesk)

WordPerfect 5.1 (WordPerfect)

Lotus 1-2-3, Symphony (Lotus)

Word 5.0 (Microsoft)

Fonts: Adobe Type Manager (Adobe), Facelift (Bitstream), Soft Type (Z-Soft)

Drivers supplied by software vendors: Paintbrush IV™ (Z-Soft), X-Windows (Interactive Systems and The Santa Cruz Operation)

For updated information regarding additional drivers, contact Cornerstone.

DualPage 120 Specifications

Display Size	19" diagonal	**VLF Low Radiation***	Optional
Screen Area	13.8" (H) x 10.8" (V) (351 mm x 274 mm)	**Shielding: SEMKO MRP II**	
		Tilt & Swivel	Standard
Antiglare Filter	Optional	**Height**	18.5" (470 mm)
Resolution	1600 (H) x 1280 (V) pixels	**Width**	18" (457 mm)
Dots Per Inch	120	**Depth**	16.5" (419 mm)
Refresh Rate	76Hz	**Weight**	
Video Bandwidth	262MHz	**Net**	49 lbs (22 kg)
Horizontal Scan Frequency	105KHz	**Shipping**	62 lbs (27.8 kg)
Scanning	Non-interlaced	**Deflection**	114 degrees
Phosphor Type	P-104	**Power Requirements**	90–250 VAC
Video Signal	10 K differential ECL	**Power Consumption**	110 W maximum
Gray Levels	Monochrome, 4-shade or 16-shade (1, 2, or 4 bit planes)	**OperatingTemperature**	10°–40°C
		Footprint	11.75" (W) x 11.5" (D) (298 mm x 292 mm)
Overscanning	User selectable	**Regulatory Approval**	UL approved, CSA certified, CHHS Rule-21 CFR
Controls	Master brightness, contrast, power; located in front	**Regulatory Approvals Pending***	FCC Class B, VDE Class B, NEMKO, MRP II, VLF/ELF Shielding
Rear Adjustments	Horizontal size, horizontal center, vertical size, vertical center, tilt		

AT-bus Display Controller

Hosts	IBM PC/AT, PS/2 Model 30, or compatibles	**Address Modes**	Software Selectable
		Windowed Address	Two 16KB windows, addressed at 80000 or C8000 (jumper selectable)
Board Size	9.2" (H) x 4.2" (V)		
Video Bandwidth	262MHz		
I/O Slot	Single 8 or 16 bit	**Direct Address Mode**	Contiguous 2MB address space, addressed on any 2MB boundary above 1MB (software selectable)
Number of Bit Planes	1, 2 or 4 (monochrome, 4-shade or 16-shade grayscale)		
Power Required	2.3A at +5V; 50mA at -5V, 80mA at -12V	**Video Modes**	Full Hercules® Graphics Card (HGC) in hardware; includes 720 (H) x 348 (V) monochrome graphics mode
Extended Text Modes	96 x 40, 96 x 80 characters		
Graphics Memory	Windowed or direct		

Micro Channel-bus Display Controller

Hosts	IBM PS/2 models 50, 55, 60, 70, 80, 90, 95 and equivalent systems	**Number of Bit Planes**	1, 2 or 4 (monochrome, 4-shade or 16-shade grayscale)
Board Size	11.5" (H) x 3.5" (V) (292 mm x 89 mm)	**Address Modes**	Selectable via POS
		Emulation Supported	All VGA video modes, with monochrome translation for color modes
Video Bandwidth	262 MHz		
I/O Slot	Any slot for graphics, or video slot for VGA frame grabbing	**Power Required**	2.9 A at +5 V; 150 mA at –12 V
Graphics Memory	Two 16 KB windows	**Extended Text**	160 x 67, 110 x 67, 110 x 43 characters

*4th Quarter 1991

1990 Concourse Drive
San Jose, CA 95131
TEL: (408) 435-8900
(800) 562-2552
FAX: (408) 435-8998

800 Hingham Street
Suite 24
Rockland, MA 02370
TEL: (617) 982-8773
FAX: (617) 871-0991

Cornerstone Technology GmbH
Richard Wagner Strasse 31
8023 Pullach, Germany
TEL: 49-89-793-2043
FAX: 49-89-793-8781

Cornerstone TECHNOLOGY

MultiMode 120™ 19" display for IBM® PC and PS/2® computers

Complete with display, all required cables, and user's guide.

SOFTWARE COMPATIBILITY

Driver support is provided for these applications:

- Autodesk ADI 4.0 Display List Driver
  - AutoCAD Release 10*
  - AutoShade
  - AutoSketch
  - 3-D Studio
- Autodesk ADI 4.1 Display List Driver
  - AutoCAD Release 11* (P386)
- DRI GEM
  - All GEM applications including ...
  - Artline
  - Draw
  - 1stWord Plus
  - Graph
- IBM OS/2 Presentation Manager*
- Lotus 1-2-3
- Lotus Symphony
- Microsoft Windows*
  - All Windows applications including:
  - Adobe Illustrator
  - Aldus PageMaker
  - CorelDraw
  - Micrografx Designer
  - Microsoft Excel
  - Microsoft PowerPoint
  - Microsoft Project
  - Microsoft Word
  - Ventura Publisher
- Microsoft Word
- Ventura Publisher
- VersaCAD
- WordPerfect

And many more ...

UTILITY SOFTWARE

- MultiMode Control Panel Software
  - Resolution switching and panning under Microsoft Windows 3.x
- Menu-driven installation utility
- Diagnostics utility

DISPLAY

- CRT Size
  - 19" flat screen landscape display
- Display Area
  - 14.1" x 10.2"
- Horizontal Scan Rate
  - 94.7 kHz
- Video Interface
  - Analog
- Physical Dimensions
  - 18"H x 19"W x 17"D (height includes base)
- Phosphor
  - High efficiency, paper white
- Net Weight
  - 51 lbs / 23 kg (without OCLI), 59 lbs / 27 kg (with OCLI)
- Vertical Refresh Rate
  - 76 Hz to 116 Hz (76 Hz at 1664 x 1200 resolution)
- Input Connector
  - Male 13W3 (video cable attached to display)
- Adjustments
  - Tilt/swivel, contrast, brightness, screen rotation, horizontal and vertical centering
- Electromagnetic Emissions
  - Meets latest Swedish guidelines for reduced VLF/ELF magnetic field emissions (MPR1990:8)
  - With optional anti-glare, anti-static panel, meets all MPR1990:8 emissions guidelines

DISPLAY CONTROLLER FEATURES

- Multiple Resolutions (on-the-fly switching):
  - 1664 x 1200, 4 gray shades* — 120 dpi, 76 Hz refresh
  - 1280 x 960 — 92 dpi, 94 Hz refresh
  - 1024 x 768 — 72 dpi, 116 Hz refresh
  - 832 x 600 — 60 dpi, 76 Hz refresh
  - 640 x 480 — 46 dpi, 94 Hz refresh
  - 512 x 384 — 36 dpi, 116 Hz refresh
- All IBM VGA Modes
  - Displayed as 16 true grayscale equivalents on the MultiMode 120 display or as 256 colors on a secondary VGA display
- On-board VGA chip for 100% VGA hardware compatibility
- High performance 16-bit interface for both VGA & native graphics
- Enhanced text modes (up to 132 x 72 characters)
- VGA video connector for dual display configurations (PC version)
- Hardware screen save (patent pending)

OPERATING ENVIRONMENT

- Temperature
  - 0 to +40 degrees Celsius
- Altitude
  - Sea Level to 10,000 feet
- Humidity
  - 20 to 80% non-condensing
- Input Power — Display

	110v Units:	220v Units:
	95 to 132 VAC	190 to 264 VAC
	47 to 63 Hz	47 to 63 Hz
	90 Watts	90 Watts

REGULATORY APPROVAL

- RFI Emissions
  - FCC Class B, VDE Class B
- Safety
  - UL, TUV, and CSA Approval

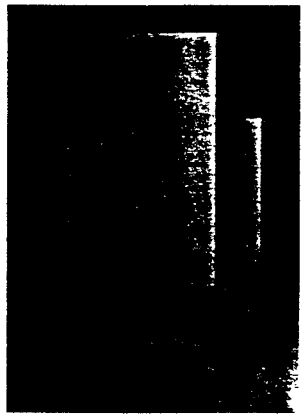
ORDERING INFORMATION

- Displays
  - MMD-00-1912A (110v)
  - MMD-00-1912B (220v)
  - MultiMode 120 display (includes all cables)
  - MMD-00-1S12A (110v)
  - MMD-00-1S12B (220v)
  - MultiMode 120 display with OCLI anti-glare, anti-static bonded panel (includes all cables)
- Display Controllers
  - MMA-PC-00020
  - MultiMode 120 display controller for IBM PC and exact compatible computers
  - MMA-PS-00020
  - MultiMode 120 display controller for IBM PS/2 Micro Channel and exact compatible computers

SIGMA DESIGNS®
47900 Bayside Parkway • Fremont, CA 94538
TEL: (510) 770-0100 • FAX: (510) 770-2640
COMPUSERVE: GO DTPVEN • BBS: (510) 770-0111 (9600-8-1-N)

*4 gray shades with Windows 3.x, AutoCAD 10 & 11 and IBM OS/2 Presentation

MM120-0392-04



CPU Subsystem Features

- 2 to 8 Intel486™ 50 MHz microprocessors
- 8 KB internal cache
- Internal floating point processor
- 128 KB external read cache
- 32 KB external write cache
- Dual 64-bit processor to memory bus
- 400 MB/sec total bandwidth
- Up to 512 MB of dual ported, 4-way interleaved RAM with ECC error detection and correction

I/O Subsystem Features

- Micro Channel® architecture – Extended dual I/O buses
- Second bus optional
- 80 MB/sec per bus transfer rate
- 32-bit bus width per slot
- 8 or 16 free bus slots
- Synchronous SCSI adapter
- 5 MB per SCSI channel
- 16 total SCSI channels
- 28 full height* fixed drive bays
- 4 full height* removable bays

**Each full-height bay may be configured with two half-height or two 3.5" drives*

Peripheral Devices

- 3.5", 1.44 MB internal flex disk
- Disk Capacity – Internal
  - 340 MB to 36 GB
- Disk Capacity – External
  - Up to 50 GB
- Tapes – Internal
  - 525 MB Quarter inch cartridge (QIC) tape (1 included with system
  - 1.3 GB 4 mm DAT tape
- Tapes – External
  - 2.5 GB 8mm tape
  - 1/2" reel-to-reel tape
- CD-ROM Int/Ext
  - 600 MB CD-ROM

Optional I/O Features

- Up to 8 LAN connections, including 4 Token Ring and 4 Ethernet connections
- Up to 12 WAN connections
- Up to 1024 async TTY connections

System Interfaces

- 2 RS-232 serial interfaces and 1 parallel printer interface for system support
- Keyboard and mouse
- Super VGA interface with 1024 x 768 pixel resolution and color

Security

- Cabinet lock
- System password
- Keyboard password

Specifications

- Physical Dimensions
  - Height: 56 inches
  - Width: 34 inches
  - Depth: 28 inches
  - Weight: 800 pounds, fully configured
- Operating Environment
  - Operating Temperature: 5°C to 45°C (40°F to 113°F)
- Power Requirement
  - Voltage Range: 200 to 240 VAC

NCR Corporation continuously improves products as new technologies and components become available. NCR Corporation, therefore, reserves the right to change specifications without prior notice.

All features, functions and operations described herein may not be marketed by NCR in all parts of the world. Consult your NCR representative for the latest information.

Intel486 is a trademark of Intel Corporation.

Micro Channel is a registered trademark of IBM Corporation.

SP 3928 0791

SPARCstation 10 Series Specifications

Model	30	41	52	54
Number of processors	One	One	Two	Four
SPECint 92	44.2	52.6	58.1 per CPU	
SPECfp 92	52.9	63.7	71.4 per CPU	
SPEC throughput 89	N/A	N/A	100*	218*
MIPS	86.1	96.2	107.3 per CPU	
MFLOPS	10.6	17.2	19.0 per CPU	
On-chip cache	36 KB	36 KB	36 KB per CPU	
SuperCache	None	1 MB	1 MB per CPU	

Processor

Architecture	Superscalar SPARC Version 8
Memory management	SPARC reference MMU with 65,536 contexts
Cache	Primary: 20-KB instruction, and 16-KB data on-chip Secondary: 1-MB external optional
CPU interface	Two 64-bit MBus slots for multiprocessing

Main Memory

16- and 64-MB SIMM expansion
128 MB maximum (with 16-MB SIMMs)
512 MB maximum (with 64-MB SIMMs)

Standard Interfaces

Ethernet	10-Mb/sec twisted pair standard (10BaseT) AUI available with optional adapter cable
SCSI	10-MB/sec SCSI-2 (synchronous)
Serial	Two RS-232C, RS-423 serial ports
Parallel	Centronics-compatible parallel port
Audio	CD-quality 16-bit audio, 8 to 48 KHz External speaker box and microphone
ISDN	Dual basic-rate (2B+D) interface 144 Kb/sec
SBus	Four expansion slots; 32-bit data bus width

Mass Storage

Floppy disk	3.5-in. MS-DOS-IBM*-compatible (720 KB, 1.2 MB, 1.44 MB formatted)
Internal disk	Up to two 3.5-in. disks (424 MB or 1 GB formatted)
External desktop storage	Disk: 424 MB 3.5-in.; 1.3 GB 5.25-in., 644-MB CD-ROM Tape: 150-MB .25-in QIC-150, 5 GB 8 mm

Graphics Options

SPARCstation 10GX	8-bit 2-D, 3-D wireframe, 1152 x 900 resolution
SPARCstation 10GXplus	8-bit 2-D, 3-D wireframe, 1280 x 1024 resolution, hardware double buffering
SPARCstation 10GS	24-bit 3-D solids, 1152 x 900 resolution, double buffering, Z-buffer Gouraud shading, depth cueing
SPARCstation 10GT	24-bit 3-D solids, 1280 x 1024 resolution, double buffering, Z-buffer Gouraud shading, depth cueing transparency, anti-aliasing, overlays

Monitor Options

16-in. color	1152 x 900 resolution, 76- or 66-Hz refresh rate, 100 dots per inch
19-in. color	1152 x 900 resolution, 76- or 66-Hz refresh rate, 84 dots per inch 1280 x 1024 resolution, 67-Hz refresh rate, 93 dots per inch (GXplus)
21-in. color	1280 x 1024 resolution, 76-Hz refresh rate, 87 dots per inch
19-in. grayscale	1152 x 900 resolution, 76-Hz refresh rate, 84 dots per inch

Input Devices

Keyboard	Sun Type 5; AT 101 or UNIX layout available More than 18 international keyboards available
Mouse	Optical, 3-button

Software

Operating system	Solaris 1.1 SMCC Version A
Windowing system	OpenWindows Version 3
Languages	C, C++, Pascal, FORTRAN
Networking	ONC, NFS* TCP/IP, SunNet™ OSI, MHS
Graphics	SunVision,™ SunPHIGS™ XGL,™ SunGKS™ Xlib, PostScript*

Environment

AC power	100-240 VAC, 47-63 Hz, 0.4 K VA
Operating	0° C to 40°C (32° F to 104° F) 5% to 95% relative humidity, noncondensing
Nonoperating	-40° C to 75° C (-40° F to 167° F) 5% to 95% relative humidity, noncondensing
Operating acoustic noise	5.1 bels (at 28° C)
Idle acoustic noise	5.0 bels (at 28° C)

Declared noise emissions in accordance with ISO 9296

Regulations

Meets or exceeds the following requirements

Safety	UL 1950, CSA 950, TUV EN 60950
RFI/EMI	FCC Class B, DOC Class B, VDE Class B, VCCI Class 2
X-ray	DHHS 21, Subchapter J, PTB German X-ray Decree

Dimensions and Weights

SPARCstation 10 chassis	
Height	7.6 cm (3.0 in.)
Width	41.7 cm (16.4 in.)
Depth	40.9 cm (16.1 in.)
Shipping weight	12.7 kg (27.0 lbs.)
16-in. color monitor	
Height	41.6 cm (16.4 in.)
Width	40.6 cm (16.0 in.)
Depth	45.3 cm (17.8 in.)
Shipping weight	27.3 kg (60.0 lbs.)
19-in. color monitor	
Height	47.4 cm (18.7 in.)
Width	48.0 cm (18.9 in.)
Depth	50.5 cm (19.9 in.)
Shipping weight	38.6 kg (85.0 lbs.)
19-in. grayscale monitor	
Height	45.0 cm (17.7 in.)
Width	46.0 cm (18.1 in.)
Depth	41.0 cm (16.1 in.)
Shipping weight	27.7 kg (61.0 lbs.)

Upgrades

Upgrades are available for the SPARCstation 1, SPARCstation 1+, SPARCstation IPX, SPARCstation 2, and Sun-3™ systems.

*Multiprocessing system performance numbers are estimates. No SPEC throughput 92 definition exists at this time for multiprocessing, so SPEC throughput 89 estimates are provided. Performance figures are generated using a combination of Apogee compilers with Kuck & Associates pre-processor and SunPro™ KAP compilers.

Sun Microsystems
Computer Corporation

2550 Garcia Avenue, Mountain View, CA 94043 415 960-1300 Fax 415 969-9131

For U.S. sales office locations, call:
800 821-4643
In California:
800 821-4642

Australia (02) 413 2666
Belgium +32 2 730 05 11
Canada 416 477-6745
Finland +358-0-502 27 18
France (1) 30 67 50 00
Germany (089) 46 00 80
Hong Kong 852 802 4188
Italy (039) 60551
Japan (03) 3221 7021
Korea 822-563-8700
Latin America 415 688-9464
Netherlands 033 501234
New Zealand (04) 499 2344
Nordic Countries +46 (0)8 623 90 00
PRC 861-831-5568
Singapore 224 3388
Spain (91) 5551648
Switzerland (1) 825 7111
Taiwan 2 514-0567
UK 0276 20444

Elsewhere in the world call Corporate Headquarters 415 960-1300
Intercontinental Sales 415 688-9000

Printed in USA 5/92 DE486-0/330K

DATABASE: Technical Documentation Library

PACKAGE: VENDORS

Attribute Name	Length	Data Type	Properties	DB Name	Prompt
Vendor Name	40	LOOKUP	Descriptio	VEN_NAME	Select Vendor from list...
Addr Line 1	80	ALLCAP		ADDR_1	Enter first line of address...
Addr Line 2	80	CHAR		ADDR_2	Enter second line of address...
Addr Line 3	80	CHAR		ADDR_3	Enter City, State(XX), Zip (nnnnn-nnnn)...
Phone #	13	CHAR		MAIN_PH	Enter phone number [(nnn)nnn-nnnn]...
800 #	8	CHAR		V800_PH	Enter 800 number (nnn-nnnn)
FAX #	13	CHAR		FAX_PH	Enter fax number [(nnn)nnn-nnn]...
Contact Name	40	CHAR		CONT_NAME	Enter name for point of contact...
Contact Title	40	CHAR		CONT_TITL	Enter title for point of contact...
Vendor ID	10	CHAR	Index	VEN_ID	(hidden system field)

LIBRARY: TECHDOC

Attribute Labe	Length	Data Type	Properties	DB Name	Prompt
Vendor Name	40	LOOKUP		VEN_NAME	Select Vendor from list...
Prod. Cat.	30	ALLCAP	Index/Desc	PROD_CAT	Enter Product Category...
Prod. Name	20	CHAR	Fast Looku	PROD_NAME	Enter Product Name...
Capacity	20	CHAR		CAPACITY	Enter Product Capacity...
Performance	20	CHAR		PERF	Enter Product Perf Characteristic...
Release	20	CHAR		RELEASE	Enter Release/Version Number...
Doc. Date	11	DATE		DOC_DATE	Enter Date Document Published (mm/dd/yy)...
Scanned Date	11	DATE		SCAN_DATE	Enter Scan Date or F8 for Today's Date...

DATABASE: Revised Technical Documentation Library

PACKAGE: VENDORSR

Attribute Name	Length	Data Type	Properties	DB Name	Prompt
Vendor Name	40	LOOKUP	Descriptio	VEN_NAME	Select Vendor from list...
Phone #	13	CHAR		MAIN_PH	Enter phone number [(nnn)nnn-nnnn]...
800 Number	8	CHAR		V800_PH	Enter 800 number (nnn-nnnn)
Vend Cntct Nam	40	CHAR		CONT_NAME	Enter name of vendor point of contact...
Vend Cntct Tit	40	CHAR		CONT_TITL	Enter title of vendor point of contact
Corp Addr 1	80	ALLCAP		ADDR_1	Enter first line of address...
Corp Addr 2	80	CHAR		ADDR_2	Enter second line of address...
Corp Addr 3	80	CHAR		ADDR_3	Enter City, State(XX), Zip (nnnnn-nnnn)...
FAX Number	13	CHAR		FAX_PH	Enter fax number [(nnn)nnn-nnn]
Int. Contact	40	CHAR		INSID_CON	Enter name of internal point of contact...
Vendor ID	10	CHAR	Index	VEN_ID	(hidden system field)

LIBRARY: TECHDOCR

Attribute Labe	Length	Data Type	Properties	DB Name	Prompt
Prod. Cat.	30	ALLCAP	Index/Desc	PROD_CAT	Enter Product Category...
Prod. Name	30	CHAR	Fast Looku	PROD_NAME	Enter Product Name...
Capacity	20	CHAR		CAPACITY	Enter Product Capacity...
Performance	20	CHAR		PERF	Enter Product Perf Characteristic...
Release	20	CHAR		RELEASE	Enter Release/Version Number...
Vendor Name	40	LOOKUP		VEN_NAME	Select Vendor from list...
Doc. Date	11	DATE		DOC_DATE	Enter Date Document Published (mm/dd/yy)...
Scanned Date	11	DATE		SCAN_DATE	Enter Scan Date or F8 for Today's Date...

REFERENCES

Association for Information and Image Management. 1992. Electronic Imaging Request for Proposal (RFP) Guidelines (AIIM TR27-1991). Silver Spring, MD: AIIM C15.1 Electronic Imaging/RFP Committee.

Avedon, Don M. 1992. Introduction to Electronic Imaging. Silver Spring, MD: AIIM.

Boehm, B.W. 1973. Software and Its Impact: A Quantitative Assessment. DATAMATION, (May): 48-59.

Boehm, B.W. 1988. Spiral Model of Software Development and Enhancement. IEEE Computer, 21(5): 61-72.

Bowles, Adrion J. 1990. A note on the Yourdon Structured Method. Software Engineering Notes, 15(2): 27.

Cinnamon, Barry, and Richard Nees. 1991. The Optical Disk ... Gateway to 2000. Silver Spring, MD: AIIM.

Coad, Peter, and Edward Yourdon. 1991a. Object-Oriented Analysis. Englewood Cliffs, NJ: Yourdon Press/Prentice-Hall.

Coad, Peter, and Edward Yourdon. 1991b. Object-Oriented Design. Englewood Cliffs, NJ: Yourdon Press/Prentice-Hall.

Cobb, Richard H., and Harlan D. Mills. 1990. Engineering Software Under Statistical Quality Control. IEEE Software (November): 44-54.

Constantine, Larry L. 1989. Methodology: The Experts Speak "The Structured-Design Approach". Byte, 14(4): 232-233.

Cutts, Geoff. 1991. Structured Systems Analysis and Design Methodology. Oxford: Blackwell Scientific Publications.

DeGrace, Peter, and Leslie Hulet Stahl. 1990. Wicked Problems, Righteous Solutions: A Catalogue of Modern Software Engineering Paradigms. Englewood Cliffs, NJ: Yourdon Press/Prentice-Hall.

DeMarco, Tom. 1978. Structured Analysis and System Specification. New York: Yourdon, Inc.

Fichman, Robert G., and Chris F. Kemerer. 1992. Object-Oriented And Conventional Analysis and Design Methodologies. IEEE Computer, 25(10): 22-39.

Hammer, Michael. 1990. Reengineering Work: Don't Automate, Obliterate. Harvard Business Review, July-August: 104-112.

Henderson-Sellers, Brian, and Julian M. Edwards. 1990. The Object-Oriented Systems Life Cycle. Communications of the ACM, 33(9): 142-159.

Hetzel, William. 1988. The Complete Guide To Software Testing (2d ed.). Wellsely, MA: QED Information Sciences.

Hevner, Alan R. 1992. Object-Oriented System Development Methods. Advances in Computers, 35: 135-198.

Hevner, Alan R., Tomas Vagoun, and Doug Lemmon. 1992. Quality Measurements in the Cleanroom Development Process. Class handouts from Systems Analysis and Design course. University of Maryland, Fall Semester.

Killen, Michael. 1991. SAA Image Processing. New York: McGraw-Hill, Inc.

Korson, Tim, and John D. McGregor. Understanding Object-Oriented: A Unifying Paradigm. Communications of the ACM. 33(9): 40-60.

Latimer, Paul J. 1992. Binary Vs. Grayscale Images: A Tradeoff of Storage Vs. Quality. Imaging. (July): 20-21.

Major, Michael J. 1992. Worth a thousand words: IBM's ImagePlus is bringing imaging into the mainstream. MIDRANGE Systems, 5(13): 23(3).

Marca, David A., and Clement L. McGowan. 1988. SADT™ Structured Analysis and Design Technique. New York: McGraw-Hill Book Company.

May, Thornton A. 1993. Surviving Imaging: The "Value Thing." INFORM (January): 43-46.

McMenamin, Stephen M., and John F. Palmer. 1984. Essential Systems Analysis. New York: Yourdon, Inc.

Middleton, Timothy. 1992. Mainframe bondage. Corporate Computing, 1(3): 162(5).

Mills, Harlan D., Richard C. Linger, and Alan R. Hevner. 1986. Principles of Information Systems Analysis and Design. Orlando, FL: Academic Press, Inc.

Mills, H.D., R.C. Linger, and A.R. Hevner. 1987. Box Structured Information Systems. IBM Systems Journal, 26(4) 395-413.

Mosley, Daniel J. 1993. The Handbook of MIS Application Software Testing. Englewood Cliffs, NJ: Yourdon Press.

Myers, Glenford. 1979. The Art of Software Testing. New York: Wiley-Interscience.

Safdie, Elias, Tom Flanagan, and The DMR Group. 1992. DOCUMENT IMAGING Implementation Strategies from Planning through Production. (Available from The Applied Technologies Group, a division of International Data Corporation, Five Steen Street, Framingham, MA 01701)

Short, K.W. 1991. Methodology integration: evolution of information engineering. Information and Software Technology, 33(9): 720-732.

Stadler, David A., and Stephen A. Elliott. 1992. Remake Your Business. Inform (February): 12-15, 17.

Trammell, A. Brinkley. 1992. Acceptance - Imaging's biggest hurdle. Inform (April): 25-27.

Wallace, Scott. 1992. Image Archiving. Corporate Computing, 1(4): 75-82.

Webster's New World Dictionary Second College Edition. 1986. New York: Prentice Hall Press.

Yourdon, Edward. 1992. Decline & Fall of the American Programmer. Englewood Cliffs, NJ: Yourdon Press.

Yourdon, Edward and Larry L. Constantine. 1978. STRUCTURED DESIGN Fundamentals of a Discipline of Computer Program and Systems Design. New York: Yourdon Press.

Zachman, J.A. 1987. A framework for information systems architecture. IBM Systems Journal, 26(3): 276-292.