UNIVERSITY OF ALBERTA

THE DESIGN AND IMPLEMENTATION OF A COMPUTERIZED STUDENT INFORMATION SYSTEM: A CASE STUDY

BY

RALPH LLOYD SCHIENBEIN

A thesis submitted to the Faculty of Graduate Studies and Research in
partial fulfillment of the
requirements for the degree of DOCTOR OF PHILOSOPHY.

DEPARTMENT OF EDUCATIONAL ADMINISTRATION

EDMONTON, ALBERTA

FALL 1993

ISBN 0-315-88280-8

THE UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR: RALPH LLOYD SCHIENBEIN

TITLE OF THESIS: THE DESIGN AND IMPLEMENTATION OF A COMPUTERIZED STUDENT INFORMATION SYSTEM: A CASE STUDY

DEGREE: DOCTOR OF PHILOSOPHY

YEAR THIS DEGREE GRANTED: 1993

Signed: [signature]

Permanent Address:
13 Rosswood Estates
22151 Township Road 530
Ardrossan, Alberta
T0B 0E0

Date: 27 Aug 93

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **THE DESIGN AND IMPLEMENTATION OF A COMPUTERIZED STUDENT INFORMATION SYSTEM: A CASE STUDY** submitted by **RALPH LLOYD SCHIENBEIN** in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Dr. T. C. Montgomerie

R. G. McIntosh
Dr. R. G. McIntosh

Donald M. Richards
Dr. D. M. Richards

Eric Romaniuk
Dr. E. W. Romaniuk

GK
Dr. G. Kearsley

Date: 27 Aug 93

Abstract

The development of computerized information systems occurs in many organizations. The development paradigms used within computing development groups are often determined by computer consulting firms, computer vendors, and computing science literature. This case study deals with the successful design and implementation of a computerized student information system (S.I.S.) in an education (K-12) environment.

Various aspects of the development are studied. Issues examined includes the examination of implementing change (the change model), the use of prototyping as a development paradigm, the role of a boundary spanner to serve as a linking agent between the computer programmers and various organizational stakeholders, and organizational management styles. The choice of computing language for development and the larger issue of developing applications that are "machine independent" are examined. The case study chronicles developments in the computing field and parallels the organization events that occurred within the same time frame.

Acknowledgments

There have been a number of people that have been an inspiration and help to me over the years. I would like to thank my Board of Education for granting me leave to pursue further post secondary study. In particular I would like to thank the Superintendent of Schools and the Associate Superintendent for their support and encouragement. I would like to thank Dave Wodelet and Linda Adamyk who "carried an extra load" while I was away from work.

I would like to thank Dr. T. Craig Montgomerie for the tremendous time and effort that he put in support and advise during my studies. Without Craig's encouragement I would not have been in this program or undertaken this thesis. Computing has changed a great deal since the IBM 1500 in the early 1970s!

I would like to thank those on my committee for their work, feedback, and encouragement--Dr. Don Richards and Dr. Gordon McIntosh, and Dr. E. Romaniuk.

I pause and remember the many mentors that have helped me over the years: Don Zarsky, Dr. Steve Hunka, Terry Gunderson, and Dr. N. Marchak.

I would like to thank my late father, Edwin and my mother for whom this program of studies was important.

A special thanks to my wife Lesley and children (John, Karen, and Elizabeth) for their support, patience and gentle reminders of "If I have to do my homework, so do you!"

Finally, I would like to thank my God and Saviour Jesus Christ who enables, leads, and guides me as His child.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

Overview of the Study

"By wisdom a house is built, and through understanding it is established; through knowledge its rooms are filled with rare and beautiful treasures" (Proverbs 24: 3-4).

Introduction

School districts throughout Alberta have been developing computerized systems since the late 1960s. These developments began with the large urban districts of Edmonton Public School District #7 and Calgary Public School District #19 and then progressed to Edmonton Roman Catholic Separate School District #007 and Calgary Roman Catholic Separate School District #1. In 1979, Strathcona County #20 became the first Alberta county school jurisdiction to begin using computers on a large scale. Subsequent to this, most other counties in the province have undertaken computerization to various degrees.

While the computerization process was taking place, few districts appeared to be satisfied with the systems that were developed and implemented. This trend was also noted in commercial data processing.

> The power of today's computing technology is not being used as it should be in most enterprises. Data processing is bogged down in problems. Managers are not receiving the information they require from their systems. Many decisions that should be made with the aid of computers are in fact being made with hand methods or inadequate information. Systems are so difficult to change that they often inhibit the implementation of new and important procedures that managers require (Martin, 1984, p. 3).

During the 1980s school districts continued to develop and enhance their computer system software and hardware. On May 1, 1988, *every* school in County X went "on-line" with a new student information system (SIS). The SIS software was designed in a manner very different from most systems which were previously implemented in educational settings. The development methodology, the programming language, and the file organization structures were very different from those generally utilized in traditional data processing environments. The author was the project manager for the development of this new SIS system and later became the director of the computing department.

The study reported herein is a case study of the development of this SIS system and the educational administrative issues surrounding it.

Microcomputers were used by certain personnel in each school to gain access to the central student information system which resided on the district's single central computer. Each school had access to its portion of a large central data base. Two types of student related reports were available; those reports required by a school, and those reports required by district administrators. Some district administrators (e.g., those in student transportation) required specialized use of the student information. Selected data from the student information system were electronically transmitted ("downloaded") to a stand-alone computerized transportation system for computer-assisted bus route construction.

Concurrent with the implementation of the SIS was the introduction of an electronic mail facility to allow electronic messages to be transmitted both between the district's central office and the schools and among the schools. The telecommunication infrastructure which was designed for the use of the SIS and electronic mail also made it possible to use possible other on-line applications such as purchasing, interactive access to financial information, and so on.

Problem

The major research question for this study was:

What is required for the successful design and implementation of a computer based student information system in a school district?

In order to examine this problem, a case study of a computerized student information system was conducted. Attention was given to the change processes involved in the creation and implementation of a large computerized system in County X. A number of sub-problems examined in this case study were:

1. What were the key events in the design and implementation of this system? Examples of these events include "Which programming language should be used?", "Should the software be developed in-house or should a commercial package be acquired?", "If the software is developed in-house, should new development techniques, such as Martin's concept of prototyping, be used?", "Could the design be improved by utilizing a person in a boundary spanning role?", "What data elements should be used?", and so on.

2. What were the key organizational issues? Examples of such issues include "Who were the key decision makers and influential participants in the design and implementation?", "Was the organizational structure appropriate?", and so on.

3. Did Fullan's (1982) change model adequately account for the changes in County X creating and implementing a computerized decision support system?

4. What significant issues and themes emerge from this research that are of relevance to the creation and implementation of computerized systems in educational settings?

County X now has an on-line interactive student information system. This case study of the development and implementation of this student information system could be useful to other jurisdictions on the threshold of a similar development effort. Simon described the nature of decisions changing as technology permits some decision making to become *programmed* (Pugh, Hickson, & Hinings, 1983, p. 103ff). He noted the two types of decisions as being *programmed* and *non-programmed*.

> Decisions are programmed to the extent that they are repetitive and routine or a definite procedure has been worked out to deal with them. Thus they do not have to be considered afresh each time they occur... Decisions are unprogrammed to the extent that they are new and unstructured or where there is no cut-and-dried method for handling the problem (Pugh et al., 1983, p. 105).

Simon described the use of technology first in completely *programmed* operations (e.g., accounting) under clerical support. Over time, organizations moved technologies into areas requiring more judgment and which were previously unprogrammed. These movements were in the area of middle management where previously unprogrammed areas could then be incorporated into programmed procedures. The SIS examined in this study is an example of the development of a programmed system in an area which was previously unprogrammed.

Justification

Development Paradigm

Fullan (1982), writing on the subject of change, noted that the single most important idea in change was that "...*change is a process, not an event*" (p. 41). The development of computer applications in school jurisdictions, which have taken place since the late 1960s, involves the change process. This change process has involved large amounts of time, effort, and money, and, in Alberta, this process has been largely undocumented. The process of change in administrative computing in Alberta also has not been well documented. This may be due in large part to the fact that few educational administrators have the background and training to permit them to be extensively involved with the creation and implementation of a large computerized system. Many senior administrators have authority over these activities, but few have the technical background to serve as project managers. This sometimes resulted in situations where computer departments were many years behind in development and in need of massive amounts of additional funds to complete their projects (Schienbein, 1990).

Large computer system development has evolved over the years into a highly structured methodology. Carlsen and Lewis (1979) created detailed guides to provide a systematic method for analyzing projects. They included work sheets, step-by-step guides and instructions, master checklists for data collection, input forms, types of reports required, and so on. Users were usually required to "sign off" the specifications for reports and systems before any computer programs were created or modified. The rationale for this approach was that the better defined the project was at the beginning stages, then the better likelihood of a successful project implementation. This was true in situations where computing technology and organizational needs were static. With the tremendous rate of change of both hardware and software tools and organizational needs, development methodologies other than the traditional approach described by Carlsen and Lewis are warranted.

Martin (1984) questioned the applicability of traditional development methodologies. He stated that there was a crisis in data processing which resulted from *how* applications were developed, not with the computer itself.

> The problem lies not in the machine itself, but in the methods we use for creating applications. The traditional "application development life cycle" is slow and rigid. Its methods have been cast into concrete in many organizations with standards and procedures. But in many ways the procedures are not working (Martin, 1984, p. 3).

Martin proposed that one of the possible solutions to this crisis was to utilize different development methodologies, one of which was *prototyping* (p. 60ff). Martin noted that in complex engineering problems, a prototype was created before the final product was built. This allowed for the testing of principles, ensured that the system worked, and feedback obtained that enabled the design to be modified before a great deal of money was spent on the project (pp. 47-48). Martin maintained that complex data-processing systems needed prototyping more than did engineering since there was so much to learn from a pilot operation and that many changes were likely to be made. Prototypes might help solve problems in systems not working in the way end users needed.

Large projects require participation from many different segments of an organization. A rewrite team was created that was made up of stakeholders from *all* parts of the organization. Members came from finance, senior central administrators, administrators from each of the elementary, junior high and senior high divisions, student services administrators, programmers, student transportation, school secretarial staff, etc. This team met on a regular basis to perform an analysis of needs, data element definition, selection of which elements were mandatory and which were optional, lengths of data fields, and so on. This study of the stakeholder group and the change processes used allows for an examination of a single application of Martin's prototyping model. Key components and events were identified. The success of prototyping could serve as a useful model in assisting other school jurisdictions in selecting their development methodologies.

Boundary Spanner

Tushman (1977) dealt with the need for an innovative system to gather information from and transmit information to several external areas. He cited the role of the boundary spanner in the technical innovation process (Tushman, 1977, p. 587ff). A boundary spanner is a person who "bridges" the two worlds of the technical innovation and the user community. This spanner has the role of translating contrasting organizational language and culture between the innovation group and the external groups. The SIS development utilized two boundary spanners--one was a senior high school assistant principal, seconded for two years to work with the programmers, and the other was the author. The author was in a unique position to study the spanner role both as an educational administrator and as a computer developer. The author began the project as the SIS project leader and later assumed the position of the Director of Planning and Data Services.

Development Costs

Computing in education has traditionally been sub-divided into two fields: administrative computing (e.g., financial computing--general ledger, payroll, etc.) and instructional computing. Until the mid 1980s, computing was general conducted on central mainframe computers. Microcomputer-based "single user" computing emerged during the late 1970s. Microcomputer-based computing in education followed the same path as had mainframe education computing by becoming divided into instructional and administrative computing. As the rapid rate of change of computing technology continues and the power, flexibility, user friendliness, and cost of computing continues to be more cost effective, educators need to develop *environments* which allow their systems to become more efficient and effective. As central office departments and schools become autonomous with respect to their computing applications and equipment, a coordinated approach to a computing philosophy and architecture will tend to minimize stand-alone equipment and applications that may be difficult to integrate as organizational information needs integrate and mature.

The financial and staffing costs of developing (or mis-developing) computerized systems is extremely high. It is incumbent on administrators in education to create a *climate* conducive to success and thereby minimize the developmental risks. DeMarco and Lister (1987) investigated reasons as to why computer development projects often failed. They found that twenty five percent of all projects with more than twenty five years of programmer time failed to reach completion.

> The cause of failure most frequently cited by our survey participants was "politics." But now observe that people tend to use this word rather sloppily. Included under "politics" are such unrelated or loosely related things as communication problems, staffing problems, disenchantment with the boss or with the client, lack of motivation, and high turnover. People often use the word *politics* to describe any aspect of the work that is people-related, but the English language provides a much more precise term for these effects: They constitute the project's *sociology* (DeMarco & Lister, 1987, p. 4).

Besides the potential financial loss due to mis-development, the potential loss of information or the generation of inaccurate information could result in imprecise or incorrect administrative decisions.

Development Language

There have been many types of computing hardware and software. It is normal in computing environments that new hardware and software appear on the market on a continuing basis. School districts are faced with upgrading or changing their computer systems as new hardware and software become available. The conversion of existing systems to new systems is a costly and time consuming process. Since both hardware and software will continue to evolve and change, both the design of software as well as a programming language must be flexible.

Summary

This chapter has presented an overview of the study. The thesis problem was presented along with a series of sub-problems. Justification was based upon the development paradigm used in computer development, the role that a boundary spanner could play in development of systems, the impact on development costs, and the choice of computing languages used for development. Chapter 2 of the thesis deals with the review of the literature in these areas.

Chapter 2

"You diligently study the Scriptures because you think that by them you possess eternal life. These are the Scriptures that testify about Me." (John 5:39).

Literature Review

This review draws on various areas of the literature. Emphasis is placed on four areas. These are the areas of *administrative computing*, focusing on methodologies for computerized system development; *organization structures*, focusing on structures in educational settings; *leadership*, focusing on management styles; and *change*, emphasizing change and innovation in educational settings. These four areas illustrate different, but important, aspects of the development of computerized systems in an educational environment.

Administrative Computing

Educational administrators are responsible for managing their schools or school systems. In order to perform their duties, administrators need information--data and facts upon which to base their decisions and actions. Over the past three decades many computer applications have been developed to assist educational administrators by making data more available. Examples include accounting, attendance, computer aided design, energy management, human resource management, inventory, library circulation, marks, payroll, student scheduling, transportation, and so on. These computer systems were developed first in the large urban boards, due to the significant costs of both the initial hardware purchases, as well as the cost of developing software applications. The introduction of the microcomputer has resulted in virtually every school office having a computer performing some of the applications noted above.

Kearsley (1990) noted that, while there were many computers in education, it appeared that they had little impact in the school. Kearsley found:

> The lack of impact of computers in schools can be traced to a single fundamental problem: our failure to use them appropriately. Computers were perceived as instructional media to deliver drills and tutorials; as fancy calculators for math, statistics, or accounting; or as the means to teach everyone how to write BASIC programs. They were not seen as tools for writing, problem solving, decision making, data collection, creative expression, or communication (Kearsley, 1990, p. 2).

He noted that there was a profound lack of knowledge about computers on the part of teachers and school administrators. Kearsley found that during the 1980s, school jurisdictions invested in the provision of computer courses for teachers but that there was little focus on helping educational administrators to become computer literate. Bluhm (1987) cited a study by Gangel (1983) which estimated that 95% of school based computer hardware was for instruction with 5% for administration. Petruk (1986) surveyed school principals in Alberta and found that there were an estimated 22,752 microcomputers in Alberta schools on January 1, 1986. Of these 579 (2.5%) were used in the administrators' offices. These numbers consisted only of those computers in schools and did not include district staff.

Gustafson (1985) identified microcomputer administrative uses of computers as dealing with the traditional area of student records, attendance, inventory, library, equipment, and finance. Unlike other authors, Gustafson also examined district-wide use of electronic mail, data bases, selecting and acquiring equipment, spreadsheets, word processing, and security.

There are some other areas of administrative computing that are important to educational administrators. One such area is telecommunications. Martin (1990) noted the large increase in the speeds of data communication rates. These rates have increased from 300 bits per second (bps) over voice channels up to 2 million bps in the 1980s to optical transmission speeds of many millions of bps (Martin, 1990, p.7). He predicted data storage and transmission will be a critical area of implementation. With the simultaneous transmission of pictures, voice, and the sharing of large files, the telecommunications infrastructure of a district will become a critical element in a computerized district.

Data communication is vital to the running of corporations. Just as there is a telephone on everyone's desk today, so there will be a workstation on everyone's desk in the future. The data network is the nervous system of the corporation; the workstations are its nerve endings (Martin, 1989, p. 168).

The network standards to facilitate the interconnection of computers and data bases from different vendors is strategic to the organization.

Information Systems

Bluhm (1987) described a computer-based information system needed by users to provide information to conduct their business. Educational administrators need data that are "relevant (free from bias) and repeatable (the same when viewed by others)" (Bluhm, 1987, p. 6). There were three types of information and the vertical relationship of informational requirements to each of the levels of the administrative hierarchy--operational, direction and control, and planning. Operational data included factual reporting of current operations such as student progress, attendance, grades, and so on. The next level of information, direction and control, included operational information utilized to compare desired performance with actual performance--"Did expenditures exceed budgets?", "Were class sizes different than standards?", and so on. Senior administrators required information making use of relationships and projections to establish short and long range plans. Other data were required for meeting the information needs of elected officials, parent groups, and other stakeholders. Data required by different levels for different purposes Bluhm referred to as *interrelatedness* (Figure 1).

This figure has been removed because of copyright restrictions.

Figure 1
Bluhm's Interrelated Information System Levels (Bluhm, 1987, p.8)

Since data were interrelated, there were four attributes required for any information system that utilizes these data. Bluhm identified the four attributes as-- timeliness, accuracy, relevancy, and completeness. *Timeliness* was important to enable school administrators to answer questions regarding students, faculty, and building utilization (e.g., class sizes, classroom utilization, honour roll, and failures). *Accuracy* meant the data were free of errors. This included errors due to input, incorrect processing rules, improperly followed or poorly designed procedures, equipment or processing breakdowns, and so on. *Completeness* referred to having all of the *relevant* information required by an administrator to resolve a problem. Restrictions on completeness included economic restraints to compile and maintain all the information required and time constraints. Issues dealing with relevancy included keeping obsolete, "nice to know", unrefined, or too much information. Bluhm (p. 10) cited the work of Campbell, Corbally and Nystrand (1983) in lamenting the dilemma of computers generating so many charts, analyses, predictions and simulations that the administrator was overwhelmed with information and the data hid what was important and relevant.

Administrators must view accurate, timely, reliable information as a strategic investment. Senior administrators need to take an active role in creating the environment conducive to building an interrelated information system.

> Too few superintendents, it seems, are aware of the benefits of a data support system as they make policies and allocate resources that affect the information needed for planning. Data frequently doesn't exist or is not in a format easy and convenient to use. As a consequence, superintendents need to be the catalysts in establishing data support systems (Bluhm, 1987, p. 18).

The importance of making information a strategic item was also noted by Martin. "Top management must understand the need for building the foundation stones....Information which is strategic to the running of the enterprise should be identified" (Martin, 1984 , p. 84).

Mintzberg (1989, p. 350ff) cited the work of Jelinek (1979) who studied the application of Taylor's work at Texas Instruments. Jelinek argued that Taylor's successes in the factory could be replicated in the executive offices by processes fundamentally the same as Taylor's, but at a different level of abstraction. Texas Instruments designed *information management systems* intended to capture knowledge about tasks. It was found that these systems did not capture knowledge and in fact were found to discourage innovation. Mintzberg noted that there was no evidence to indicate that Texas Instruments' success stemmed from anything other than a reliable capable leader who knew how to attract good people and invigorate them (p. 350). "Good people, of course, make for good organizations. They also design good systems, at least systems that are good for them. But remove the good people and the systems collapse. Innovations, it turned out, could not be institutionalized" (p. 350).

Human Resources

One of the most important aspects of a high degree of computer use in school districts is the presence of a highly qualified, competent data-processing director (Bluhm, 1987, p. 42). Bluhm stated that this high use of computer services was directly related to the time and money invested in the director's role. To a lesser extent, the district size had a bearing on the hiring of a director. Bluhm stated that the most important qualities of the director were inquisitiveness, interest, ability to learn quickly, willingness to work hard, and a desire to meet the needs of users (p. 42). Other areas noted were the possession of human relationship skills, technical skills, and conceptual skills.

Senior management plays an important role in computerization in school districts. Bluhm found the one factor that inhibits success of a computerized system was the characteristics and attitudes of top management (p. 43). Their attitudes set the tone of how computer services would be received and used by schools. In addition to these attitudes, their views on information were also important.

> Though a school district may have a skilled data-processing director, this person will be hampered in establishing and maintaining an effective and efficient computer information system if top management is not behind the endeavor. The same holds true for a district coordinator for instructional computing....Critical to the overall success of the computer resource, however, is the degree of positive support it receives from the organizations top administrators. In the schools this means the building principals and the superintendent. (pp. 43-44).

The importance of the superintendent and central office personnel in the change process was described by Fullan:

> All superintendents and district administrators with program responsibilities are involved in some manner with change. The variation comes into play in relation to how change is approached and reacted to, not whether change is considered. A quick review of the evidence shows conclusively how important district administrators are. What they do at each of the three main phases of change--the initial decision or mobilization, implementation, and institutionalization--significantly affects the destiny of the proposed change (Fullan, 1982, p. 162-163)

DeMarco and Lister (1987) coined the term "High Tech Illusion" (p. 5). This referred to a widely held conviction that people working in the computer industry were "in computers" or "in telecommunications" while other people used technology in their work. DeMarco and Lister noted that technology people work in teams and on projects and were actually in the human communication business instead of the technology business. The reason people focused on technology was because it was easier to focus on technology than the human side of the work.

> Getting the new disk drive installed is positively trivial compared to figuring out why Horace is in a blue funk or why Susan is dissatisfied with the company after only a few months. Human interactions are complicated and never very crisp and clean in their effects, but they matter more than any other aspect of the work....If you find yourself concentrating on the technology rather than the sociology, you're like the vaudeville character who loses his keys on a dark street and looks for them on the adjacent street because, as he explains, 'The light is better there.' (p. 5).

DeMarco and Lister subscribed to the social model of change. They stated that "Development is inherently different from production" (p. 7) and that different rules applied to the development segment of software project than in the production segment. Using a fast food franchise as an example, efficient production measures were cited: squeeze out errors, take a hard line on workers, treat workers as interchangeable pieces of a machine, standardize procedures, eliminate experimentation, etc. In managing *thinking workers*, opposite measures were required such as treating people as unique, not thinking of people as parts of a machine, etc.

Martin (1984, p. 266) coined the term *intellect-intensive society* as the destination of advanced countries. In this society industry will remain vital, but many intellect workers would not be information workers. Many aspects of work in an intellect-intensive society would be automated. Workers not involved with intellect-intensive jobs would have jobs that would involve "human warmth, craftsmanship, and creativity" (p. 266). Martin listed the aspects that could not be automated as originality, intuition, inspiration, art, leadership, salesmanship, humor, love, and friendship (p. 264ff). Martin held that one of the most important aspects of the introduction of high technology was that it went hand in hand with higher levels of human interaction such as care, love, teaching, etc. (p. 265). To expand the technological potential requires a corresponding expansion of the human potential. The development of user friendly systems requires developers who have the human aspects listed.

Prototyping

The creation of a working model or prototype for end user feedback was described as being an important development paradigm. Martin (1984, p. 46ff) described the need for prototyping in the creation of complex data-processing systems. These "pilot operations" take into account that the solutions to system problems may not work in the way required by the end users. Martin envisioned a systems analyst working directly with an end user. The analyst creates data-base queries, reports, and screens that are demonstrated to the end user. The analyst then discusses and obtains feedback from the user. The working model becomes the focus of a debate that ensures that both the user and analyst are talking about the same thing. This dialogue is continued until the users are satisfied that the material meets their perceived needs. This process is in contrast with the traditional, highly structured approach as outlined by Carlsen and Lewis (1979).

Prototyping illustrates *adhocracy*, where normal rules are temporarily put on hold and new, previously unused methodologies are contemplated and used. Many computer development projects have an element of "Skunkworks" (Martin, 1984, p. 254) resulting from users saying "Well, what about... "

> It is easier, faster, and much more satisfactory for an analyst to build a prototype, given the right tools, than to work out detailed written specifications...one highly complex financial management system went through six levels of prototyping...almost certainly could not have been created with COBOL. (Martin, 1984, pp. 192, 193).

Knowledgeable users, working with data professionals utilizing newer tools in a tightly coupled group and combined with a loosely coupled stakeholders' group and management support, have produced systems that could not have been produced in a "traditional" environment. Any good decision support system "is never finished" and must continue to grow. To this end, a portion of a programmer/analyst must continue to be assigned to this application beyond the maintenance level.

Martin (1984, pp. 225-226) wrote about computer development that was "divorced from reality." He noted that

> There are increasing numbers of intellectuals who do research, write papers, or assemble ideas which are unlikely to be translated into reality. These people often believe that they are in the real world when they are far from it. Their ideas are never customer tested. One single phrase in these groups can represent complex sets of ideas, but when they hear such phrases from other departments explained in detail, they nod politely without understanding. (p. 225).

The inability of management to verbalize a phenomenon usually resulted in their not thinking about it or discussing it. Bureaucrats were cited as a form of resistance to change in organizations which were developing computer systems. "Computing appeals to graduates who want to live in an intellectually rich environment such as that of their universities, removed from grubby considerations of making a profit. White-collar automation appeals to bureaucrats, who create rigid procedures and forms to fill in" (Martin, 1984, pp. 225-226). Martin further noted that "Unfortunately, computers and office automation both act as powerful amplifiers of these activities" (p. 225). Internal intergroup conflict often results from the creative development process competing with bureaucratic tendencies.

The world of computers is turbulent in its rate of change. "Microprocessors are plunging in cost. We know that we can produce much more powerful 32-bit microprocessors..." (p. 227). Already considerably faster microcomputers (e.g., PCs with clock speeds of over 50 MHz versus the original IBM PC, at 4.77 MHz, as well as the next generation of 80486 CPU's) are on the market. It is worth noting that the Macintosh was announced in 1984, the year of Martin's quote. Since that time there have been a number of generations of hardware from the original 128k Macintosh to the current 68040 based computers. These newer versions of the hardware and almost all of the software have appeared *since* Martin's quotation. Education specific software, (e.g., boundary analysis and enrollment analysis software) are also appearing. This type of specific software illustrates how a number of non-programmed tasks envisioned by Simon are becoming programmed.

DeMarco and Lister (1987, pp. 8ff) proposed *iterative design* to establish an environment where errors were possible. Some software designs were intrinsically defect-prone and ought to be rejected instead of being repaired. They found that many computing managers felt that it would be politically unacceptable to "throw away a product that our company has paid to produce" (p. 8). By fostering an atmosphere that did not allow for error resulted in defensiveness in the programming staff and the imposition of rigid development methodologies. The highly structured rules and procedures described by Carlsen and Lewis (1979) cannot work in a prototyping environment. Bluhm (1987) described the initiation stage of computer growth as being low in the degree of management control with loose budgets. Prototyping requires a level of *adhocracy*.

A prototype is a working model programmed for the users to test the system design principles, to ensure that the system works, and to obtain feedback from the users to enable the design to be modified. An analyst working with an end user can create and demonstrate dialogues for data-base queries, screen manipulations, report generation, and so on. The interactive discussion between the analyst and the user modifies the prototype. The discussion and modification of the prototype is continued until both the analyst and user agree as to what the software design goals should be. Depending upon which tools are used to build the prototype and the complexity of the problems, revisions can vary from a few hours to a few days or weeks.

Assumptional Analysis

While student record systems have been used for a number of years, the objectives for outcomes of such systems are difficult to define. When utilizing traditional computing development, problems must be defined in great detail *prior to* beginning any actual programming. If prototyping is to be utilized as a development methodology, then other techniques must be employed to determine outcomes. Assumptional analysis can be used to determine these intended outcomes. Assumptional analysis is useful for projects that are ill defined.

> In many respects assumptional analysis is the most comprehensive of all problem structuring methods, since it includes all procedures used in conjunction with other techniques and may focus on groups, individuals, or both. The most important feature of assumptional analysis is that it is explicitly designed to treat ill-structured problems. (Dunn, 1981, p. 130).

Dunn (1981) proposed a phased procedure to perform assumptional analysis. The first phase, stakeholder identification, involved the identification, ranking, and priorization of groups or individuals affected. The stakeholder ranking and priorization was based on an assessment of the degree to which they influence or are influenced by the process. While Dunn wrote about examining ill-structured policy problems, these procedures could also be used to define outcomes of a computer application development.

Fullan (1982) noted that there were ten assumptions basic to a successful approach to educational change:

1. Do not assume that your version of what the change should be is the one that should or could be implemented...
2. Assume that any significant innovation, if it is to result in change, requires individual implementers to work out their own meaning...
3. Assume that conflict and disagreement are not only inevitable but fundamental to successful change...
4. Assume that people need pressure to change...
5. Assume that effective change takes time...
6. Do not assume that the reason for lack of implementation is outright rejection of the values embodied in the change...
7. Do not expect all or even most people or groups to change...
8. Assume that you will need a plan which is based on the above assumptions and which addresses the factors known to affect implementation...
9. Assume that no amount of knowledge will ever make it totally clear what action should be taken...
10. Assume that change is a frustrating, discouraging business..." (Fullan, 1982, pp. 91,92)

Assumptional analysis provides for a methodology to address ill defined problems in a change environment.

Organizational Topologies

Organizations are made up of many components. Resources are allocated to these components and are reallocated over time. The structure of an organization influences the work that needs to be completed, who is to perform that work, and the resources allocated to the various parts. "Organizational structure is the manner in which an organization divides its labor into specific tasks and achieves coordination among these tasks" (Johns, 1988, p. 503). Johns described the vertical division of labor, concerned primarily with apportioning authority for planning and decision making, and the horizontal division of labor which involved the grouping of basic tasks that must be performed into jobs and then into departments so that an organization can achieve its goals. The division of labor is greatly impacted by the organizational structure. "It is entirely possible to conceive of a firm or institution that has well-motivated individual members and properly led groups and still fails to fulfill its potential because of the way their efforts are divided and coordinated" (p. 502). The structure and location of the department or group performing computer development can be of great importance to the eventual success or failure of the project.

Mintzberg (1989) wrote that there were six basic parts to an organization: operating core, strategic apex, middle line, technostructure, support staff, and ideology or culture (see Figure 2). The people who performed the work or rendered the services were the *operating core*. The system was overseen by at least one full-time manager at the *strategic apex*. As organizations became larger, more managers were needed. A *middle line* was created whose authority lay between the operating core and strategic apex. As organizations became more complex, analysts who formed the *technostructure* were required. Internal services such as cafeteria, mailroom, and public relations were performed by *support staff*. *Ideology* or culture encompasses the traditions and beliefs of an organization. Within the organization are full-time employees who are influencers. These form an *internal coalition* (p. 98). People outside the organization who seek to affect decisions and actions taken within the organization form an *external coalition*.

This figure has been removed because of copyright restrictions.

Figure 2
Six Basic Parts of an Organization (Mintzberg, 1989, p. 99)

Coordinating Mechanisms

There are two fundamental and opposing requirements within any organizational structure--division of labor and coordination. "The structure of an organization can be defined simply as the total of the ways in which its labor is divided into distinct tasks and then its coordination achieved among those tasks" (pp. 101-102). Mintzberg described six coordinating mechanisms to coordinate work:

1. *Mutual adjustment.* Coordination by informal communication.
2. *Direct supervision.* Coordination by one person issuing orders or instructions to several others whose work interrelates.
3. *Standardization of work processes.* Coordination by specifying the work processes of people carrying out interrelated work.
4. *Standardization of outputs.* Coordination by specifying the results of different work.
5. *Standardization of skills and knowledge.* Coordination by the training the workers received (e.g., surgeon or anesthetist).
6. *Standardization of norms.* Coordination by standardizing norms of the organization.

The six mechanisms were the "glue" that hold organizations together. As organizational work becomes more complex, the means of coordination shift from mutual adjustment to direct supervision to standardization before reverting back to mutual adjustment. Mintzberg noted that many organizations favored one mechanism over the others, especially at certain stages of their lives. Organizations that did not favor any one tended to become politicized as the conflicts that arose caused people to vie for influence.

Organization Design Parameters

Mintzberg viewed the essence of organizational design as the "manipulation of a series of parameters that determine the division of labor and the achievement of coordination" (p. 103). He proposed nine parameters--some of which dealt with the design of the superstructure and others that dealt with the design of the decision-making system. These parameters were: job specialization, behavior formalization, training, indoctrination, unit grouping, unit size, planning and control systems, liaison devices, and decentralization.

In addition to these parameters, there were four "situational" factors (pp. 106ff) that influenced the choice of the design parameters:

1. *Age and size of the organization.*

    - The older an organization, the more formalized was its behavior...
    - The larger an organization, the more formalized its behavior...
    - The larger an organization, the more elaborate its structure and more specialized its jobs and units...
    - Structure reflects the age of the industry from its founding (p. 106).

2. *Technical system.* The technical system referred to the instruments used by the operating core to produce outputs. "Technology" referred to the knowledge base of the organization.

    - The more regulating the technical system, the more formalized the operating work and the more bureaucratic the structure of the operating core...
    - The more complex the technical system, the more elaborate and professional the support staff...
    - The automation of the operating core transforms a bureaucratic administrative structure into an organic one (p. 107).

3. *Environment.* Environment referred to characteristics outside the organization's context (e.g., market, political climate, economic conditions, etc.).

    - The more dynamic an organization's environment, the more organic its structure...
    - The more complex an organization's environment, the more decentralized the structure...
    - The more diversified an organization's markets, the greater propensity to split it into market-based units, or divisions...

- Extreme hostility in its environment drives any organization to centralize its structure temporarily (pp. 108,109).

4. *Power.*

- The greater the external control of an organization, the more centralized and formalized its structure...
- A divided external coalition will tend to give rise to a politicized internal coalition and vice versa...
- Fashion favors the structure of the day (and of the culture), sometimes even when inappropriate (p. 109).

Configurations

By altering the proportions of the six basic parts, Mintzberg described the following seven configurations of organizations: entrepreneurial, machine, professional, diversified, innovative, missionary, and political (p. 110). School districts followed the professional organization (Figure 3). In the professional organization there was a large operating core and a reasonably large support staff. Both the middle line and technostructure were small. This type had a bureaucratic yet decentralized structure dependent on training to standardize the skill of the professionals (pp. 174ff).

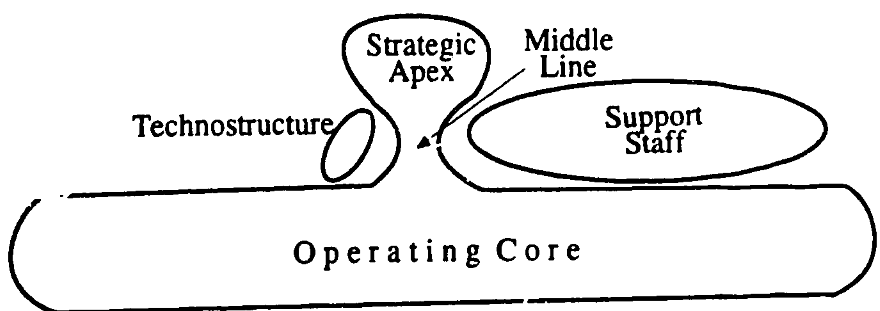

Figure 3
Professional Organization (Mintzberg, 1989, p.174)

The professional organization differed from a machine bureaucracy by emphasizing the authority of a professional nature--the power of expertise. One of the key functional characteristics was the process of *pigeonholing* (p. 177). Pigeonholing was the process of building and maintaining categories. It involved the application of a repertoire of skills (management by objectives, management information systems, etc.) to situations or contingencies that arose. Professionals demonstrated two basic tasks: to categorize or "diagnose" and to apply or execute a standard program. Mintzberg maintained that the process of "pigeonholing" differentiated professional organizations from machine or innovative organizations.

A support staff component exists within organizations. Unlike the professional component that behaved in a largely democratic fashion, the support staff were undemocratic machine-like enclaves within the professional organization. "Thus, what frequently emerges in the professional organization are parallel and separate administrative hierarchies, one democratic and bottom-up for the professionals, a second machinelike and top-down for the support staff" (p. 179).

Organizational Issues

Bluhm (1987) noted that school administrators should periodically evaluate the impact of computers in their district on their organizational structure, management control, funding, personnel and applications (p. 25). The identification of areas of organizational concern was the reason for this periodic review. The recognition that computer expansion goes through a sequence of stages gave a framework for superintendents and principals to place their problems in perspective and recognize problems as they moved towards computerization.

Bluhm identified four stages of growth--initiation, expansion, formalization, and maturity. These were very similar to Fullan's model (Figure 7). Associated with each stage was an informal organizational process that played an important role in issues resolution if a crisis was to be avoided. A major finding was that both funding and management control followed an S-shaped curve when plotted over the four stages--from a low degree of management control and a loose budget in the initiation stage to a high level of control and a tight budget in the maturity stage.

Critical to the success of a computer information system in a school district was the organizational structure. Bluhm (1987) noted that "A matter of central concern to superintendents and principals is 'Which side of the house shall have direct responsibility for administering the computer resource of the district'" (p. 28). Since early applications development were largely financial (e.g., payroll, finance, and general ledger), the treasurer was usually responsible for the computer environment. These business management personnel tended to control the priorities in computer-use applications. Problems occurred when business related reports needed to be run at the same time as a fiscal report (e.g., a payroll, general ledger month end report, and final report cards are all run at the end of June). Principals knew that payroll programs must be run on time, but also knew that student marks needed to be submitted for students to be admitted to post secondary institutions.

When problems related to computer usage increased, principals and department heads began to purchase their own computers and assigned personnel to be responsible for managing their equipment. This often resulted in the costly expenditure of funds for computing equipment and personnel. This resulted from loose budgeting practices and lax management control procedures typical of the initiation stage. To address this problem, Bluhm (1987) proposed an organizational pattern that served all areas of a district's programs (Figure 4).

This figure has been removed because of copyright restrictions.

Figure 4
Division of School District Data-processing Services (Bluhm, 1987, p. 29)

Within this division it was possible to provide the flexibility necessary to serve all aspects of the district's fiscal and student-related services with a minimum of conflict on the control and allocation of resources. Two patterns were proposed (both made the data-processing director directly responsible to the superintendent)--the formal or mature stage and the expansion stage (Figure 5).

This figure has been removed because of copyright restrictions.

Figure 5
Organizational Options (Bluhm, 1987, p. 30)

The expansion stage was characterized by the proliferation of broader and more advanced applications and an increased number of personnel becoming more specialized in the use and programming of the computer. Lacking were both clear management guidelines for project priorities and informal relationships between and among the computer personnel and users. Bluhm cited the work of Gibson and Nolan (1983) who stated "It is a period of contagious, unplanned growth, characterized by growing responsibilities for the EDP (electronic data processing) director, loose (usually decentralized) organization of the EDP facility and few explicit means of setting project priorities or crystallizing plans" (p. 31). For this environment the expansion stage organization structure was best suited. Bluhm further noted that microcomputers appeared to currently be in the expansion stage since the emphases continued to be on hardware acquisition and writing or buying software, little systematic planning, and inequities between the "have" and "have not" schools due to the financial ability or inability of districts to purchase machines.

As administrators became more aware of the effects of uncontrolled growth on their budgets, organization controls were often implemented which could result in this stage ending in crisis. This was typified by such questions as "Can the data processing effort be afforded?" and "Are our needs being met?" (p. 33). To answer these questions, information needed to be compiled on costs, operating procedures, types of programs, project priorities, personnel assignments, and so on. Control measures were the result of the efforts to answers those questions. Bluhm suggested that management techniques ordinarily utilized in the formalization stage needed to be implemented no later than the onset of the expansion stage. Such organizational techniques included the reorganization of the data-processing department, justification of the computer services budget, organization of priority setting steering committees, implementation of a management reporting system, and assignment to senior managers of responsibility for monitoring and evaluating data-processing services. The focus shifted from the computer to managing data resources. The organizational structure most effective for the mature stage was a flatter structure (Figure 5).

The aspect of different "enclaves" with different processes within an organization was not well described in Bluhm's model (Figure 4). In order to make use of the loose management control and budget described by Bluhm, it is necessary to have the innovating department developed more along the lines of Mintzberg's innovative organization. In the context of computer development, the organizational *style* of a computer department needs not only to be that outlined by Bluhm, but must operate along the lines of an innovative organization on some time certain basis. Mintzberg maintained that "None of the configurations so far discussed is capable of sophisticated innovation. ..." (p. 198). To be innovative, the organization "must avoid all the trappings of bureaucratic structure, notably sharp divisions of labor, extensive unit differentiations, highly formalized behaviors, and an emphasis on planning and control systems. Above all, it must remain flexible" (p. 199). Sophisticated innovation required *adhocracy* to hire and give power to experts and not rely on classical principles of management.

> Adhocracy is a term coined by Warren Bennis a number of years ago, and it's an attempt at saying something that is polar-opposite to the bureaucracy. Unlike the bureaucracy, where everything is laid down, adhocracy is more of an amoebic sort of process of adapting as you go along (Morgan, 1987, Financial Post).

> The adaptively-structured organization, or Adhocracy, 'is not structured to screen out heterogeneity and uncertainty'. It adapts to it's client's individual problem rather than trying to fit it into one of its own categories (Mintzberg, 1979, p. 354).

Both Morgan (1987) and Mintzberg (1989) pointed out the need for release from the normal bureaucratic environment to one where things can adapt and change. This is the climate which must be created by senior management. Mintzberg described two types of adhocracy in an innovative organization--operating and administrative. Operating adhocracies existed to serve its clients while administrative adhocracies existed to serve itself by bringing new facilities or activities on line. The relationship between the operating core and the administrative component in an adhocracy was unlike any other configuration (p. 204). The need for traditional direct supervision is diminished.

Leadership: Developing Human Resources

The creation of a computerized system involves change and is influenced by an organization's structure. Another aspect of developing these systems is the area of leadership. Leadership is required both to carry the planned change through an organization as well as for the programmers and staff creating the system. A great deal of literature has been devoted to leadership and situational approaches to leadership. Situational approaches require that the leader behave in a flexible manner, diagnose the leadership style appropriate to the situation, and apply the appropriate style. Tannenbaum and Schmidt (1957) proposed a continuum of leader behavior. In this theory there were two ends to the continuum--democratic and authoritarian. Concern for tasks was associated with the authoritarian while concern for relationships was associated with the democratic. Authoritarian leadership was based on the assumption that the power of leaders was derived from that position that they occupy and that people were innately unreliable and lazy -- Theory X (Hersey & Blanchard, 1988, p. 107). Democratic leadership assumed that the power of leaders was granted by the group that were to be led and that people could be basically creative and self-directed at work.

The Contingency Theory of Leadership developed by Fiedler (1967) suggested that there were three major situational variables that determined if a given situation was favorable to leaders: leader-member relations, the degree of structure in a group task that had been assigned, and the power and authority that the leader position provided. The favorableness of the situation was the degree to which a situation enabled a leader to exert influence over their group. There were eight combinations or situations of these three variables. The most favorable situation occurred when leaders had good leader-member relations, high position power, and a well defined job. The most unfavorable situation occurred when the leader was disliked, had little positional power, and had an unstructured task. Fiedler concluded that

1. Task-oriented leaders tend to perform best in group situations that are either very favorable or very unfavorable to the leader.

2. Relationship-oriented leaders tend to perform best in situations that are intermediate in favorableness (Hersey & Blanchard, p. 109).

The House-Mitchell (1960) path-goal theory (p. 109) dealt with how leaders influenced their subordinates' perceptions of their work goals, personal goals, and paths to goal attainment. Leaders who performed best supplied what was missing from a situation. In an unstructured task situation the leader supplied the directiveness. High leader task behavior was most effective in two situations: where there were highly structured tasks and followers had a strong need for achievement and independence with a high level of education and/or experience or where the tasks were unstructured and followers had weak needs for achievement/independence and low education and experience. Low leader task behavior was most effective where followers' tasks were highly structured with followers sharing weak needs for achievement and independence with adequate levels of education and experience or followers tasks were unstructured with strong needs for achievement and independence with high education and experience.

Vroom and Yetten's (1964) Contingency Model (p. 112) was based upon the assumption that situational variables interacted with personal attributes or characteristics of a leader resulting in leader behavior that affected organizational effectiveness. This model assumed four situational variables: followers, time, and job demands interact with personal attributes; leader experience and/or communication skills resulted in leader behavior; directive style of leadership influences organizational effectiveness; and external situational variables outside of the control of the leader (e.g., world economic conditions, etc.). The manager diagnosed their situation by asking a series of seven questions. The first three questions dealt with the quality or technical accuracy of a decision, the last four concerned the acceptance of the decision by the group members.

Hersey and Blanchard (1988) proposed the Tri-Dimensional Leader Effectiveness Model (p. 116ff) in which *task behavior* and *relationship behavior* determined the following four basic quadrants of leadership behavior: high task and low relationship; high task and high relationship; high relationship and low task; and low relationship and low task (p. 116). Task behavior was the extent to which leaders were likely to organize and define the roles of group members and how tasks were to be accomplished. It established well-defined patterns of organization, communications, and how a job was to be accomplished. Relationship behavior referred to the likelihood of leaders maintaining personal relationships between themselves and group members. It included channels of communication, providing sociometric support, etc.

The third quadrant in this model dealt with effectiveness. When a leader's style was appropriate to a given situation, it was deemed to be effective. When it was inappropriate to the situation, it was deemed to be ineffective. The difference between effective and ineffective leadership was not the actual behavior of the leader but whether it was appropriate to the environment in which it was used. It was the interaction of the style with the environment that resulted in the degree of effectiveness or ineffectiveness. Effectiveness/ineffectiveness was a continuum. No single ideal leader behavior style was appropriate in all situations.

Hersey and Blanchard distinguished between management and leadership. Management was a special kind of leadership where accomplishing organizational goals was paramount. Leadership was an attempt to influence. In dealing with effectiveness, management dealt with organizational goals and leadership dealt with individual goals (p. 128). Success and effectiveness were also distinguished. Success dealt with how an individual or group behaved. Effectiveness described an internal state or predisposition of an individual or group and was attitudinal in nature. Individuals interested only in success tended to emphasize *positional power* and utilize close supervision. To be effective required more general supervision and depended upon personal power. Positional power is "top - down" in organizations while personal power is "bottom up" through follower acceptance. "Successful versus effective framework is a way of evaluating the response to a specific behavioral event and not of evaluating performance over time" (p. 130).

It is important to look at effectiveness of leadership style on the organization over a period of time. Hersey and Blanchard cited the work of Likert (p. 131ff) who identified three variables for organizational effectiveness:

1. *Causal variables* are those factors that influenced the course of developments within an organization and its results. These independent variables could be altered by the organization and were not beyond the control of the organization. Examples of these were leadership strategies, management decisions, policies, organization structure, etc.

2. *Intervening variables* are the causal variables which affect human resources (intervening variables). These variables represent the current condition of the internal state of an organization. They were reflected in the commitment to objectives, motivation, morale of members, communications, conflict resolution, etc.

3. *Output or end-result variables* are the dependent variables reflected the achievements of the organization. Organizations often only look at outputs alone (e.g., net profits, number of publications, etc.).

The relationship between the three classes of variables was viewed as stimuli (causal) acting upon the organism (intervening) which created responses (output). The level or condition of the intervening variables was produced largely by the causal variables which influenced the end-result variables. Organizations could not attempt to modify the intervening variables with as much success as modifying them through their causal variables. Long-term goals were a result of building and developing the intervening variables. Hersey and Blanchard noted that most managers tended to be promoted based on short-term output without concern with the long-term (p. 232).

Managers and leaders had to be concerned with both output variables and intervening variables. In order to develop human resources, it was necessary to devote time to long-term organizational goals as well as the short-term ones. Time is devoted to nurturing leadership potential, motivation, morale, climate, commitment to objectives, decision-making, etc. of the people in the organization. Managers had to develop the task-relevant readiness of their followers (p. 229). Hersey and Blanchard cited the findings of Likert that found "employee-centered supervisors who use general supervision *tend* to have higher producing sections than job-centered supervisors who use close supervision" (p. 230).

To influence human resource development, Hersey and Blanchard stated that managers must identify what area of a subordinate's job was to be influenced (their goals and objectives). Once these objectives were identified and understood, managers could state what constituted good performance in each area. Before beginning any developmental cycle, the manager had to decide how well a person was doing at the present time--their *readiness*. This was accomplished by asking the person or by observing the person's behavior. The appropriate leadership style was determined by a combination of the task-relevant readiness of the follower (Figure 6).

This figure has been removed because of copyright restrictions.

Figure 6
Determining Leadership Style (Hersey & Blanchard, 1988, p. 237)

Four different leadership styles (S1 - S4) could be utilized, depending upon the follower readiness. For example, assume a manager diagnosed the environment and found the task-relevant readiness of a staff member was low (R1). If the manager wanted the staff member to perform well without supervision, the manager had to determine the appropriate leadership style for beginning the developmental cycle. This style was determined by constructing a right angle from the readiness point continuum to the curved style-of leader portion of the model (S1). S1 was a high task and low relationship style. The implication to the manager would be to tell the staff member in detail what was required for the task - a "telling" style. The staff member would be told what to do then shown how to do it. The style is high on direction and low on support. If the manager wanted to increase the task-relevant readiness of the follower, the manager must be willing to take a risk and delegate some responsibility. In taking a risk in the developmental cycle this risk must be kept to a reasonable level. Since people do not learn everything the first time, the manager must repeat the process by "successive approximations" (p. 238). In so doing the follower would take successively more responsibility for performing a specific task.

Hersey and Blanchard's work demonstrated to educational administrators that various management styles must be utilized, depending upon the readiness of the follower. In this model, time was a function of the complexity of the job being performed and the performance potential of the individual and group. It was important that managers develop the long-term view of employee development.

Change

Organizations are constantly undergoing change. Havelock (1973) defined change as "any significant alteration in the status quo" (p. 4) and innovation as "any change which represents something new to the people being changed" (p. 4). Rosenblum and Louis (1981, pp. 250-260) stated that organizational change falls into the two areas of *rational* systems and *natural* systems. In rational systems the process of change is a rational *process* (p. 22) conducted in a deliberate, rational approach based on need, facts, and insight. Natural systems involve the non-rational factors of organizations such as beliefs, practices, and organizational structures. Natural emphasized non-rational elements and "assumes change is not totally predictable, but is considered a negotiated process involving mutual adaptations of the innovative objectives" (p. 22).

Hanson (1979) dealt with the *rate* at which innovation was implemented. There were three rates of implementation: planned, evolutionary, and spontaneous. Spontaneous change arose as a result of natural circumstances or random events. Evolutionary change occurred slowly over time as a result of adaptation to internal and external organizational factors. Planned change was deliberate and directed. Hanson wrote:

> ...our analyses provide considerable support for an integrated systems model, in which no particular system element necessarily dominates others in terms of its impact on the behavior of the system, its outcomes or outputs (p. 257).

There are various models for change. Johns (1988, pp. 580, 581) outlined the work of Lewin of unfreezing, change, and refreezing. This model was a sequence of organizational events occurring over time. Unfreezing occurred when there was a recognition that some current state of affairs was inadequate. This could be a structure, task, or technology. Crises were likely to stimulate unfreezing. Change occurred when some program or plan "moved the organization and/or it members to a more adequate state" (p. 581). When change occurred, the new behaviors, attitudes, or structures had to be refrozen to become permanent.

Kiesler and Sproull (1987) utilized the social interaction model and wrote the following on social change and computing:

> In order to understand what computers mean for organizations we have to see what social changes are taking place, not just what the technological changes are. Introducing more or different computers into an organization is equivalent to introducing new ways to do things (p. 28)

Kiesler and Sproull stated that to introduce new technology or modify old technology required change in three areas: resources, behavior, and attitudes. Changing resources meant technology and creating its infrastructure--allocations of time and money, space, organizational units, etc. Changing behavior meant learning to use the new technology as well as supporting and fostering new technology and introducing it in specific areas. Changing attitudes meant coming to believe that the new technology was instrumental to one's work and life (p. 30). These changes took place as part of ongoing social interactions.

Kiesler and Sproull (p. 31) identified four key social processes for technological change

1. *Routine-driven change* assumes very few major organizational changes happen all at once. Routines for allocating resources, hiring staff, renovating space, and coordinating staff are routines that must be utilized both by managers trying to invoke change as well as by managers managing the status quo.

2. *Competence multipliers*. There is a tendency for people to get better at what they practice and to practice what they are competent at doing. While any initial change as a result of implementing computers may be targeted equally at all parts of the organization, interest and circumstance result in some people spending more time on an innovation than other people. Over time, the organization becomes increasingly differentiated with respect to expertise and experience with the innovation.

3. *Solutions looking for problems*. Through competence multipliers, a given innovation gets utilized by competent partisans as a solution to whatever the problem of the day was (p. 32). Because they are competent, their solutions often receive attention. Each new adopted solution increases their competence and also increases instances of that solution in the organization.

4. *Mutual transformation of innovation and organization*. Initial innovations are an unstable phenomenon since change is driven by routines, competence multipliers, and solutions looking for problems. As an innovation moves through an organization, it changes shape and purpose. Kiesler and Sproull cited the example of an office that bought a computer for secretarial work. The improved output quality of letters and documents sometimes resulted in managers trying to produce their own letters or reports. By so doing, the innovation entered the management function.

This view of change is an organic view of the change process. It did not give a dominant place to an organization's managers or leaders other than to have them provide the conditions necessary for change--"slack, expertise, and zeal" (p. 33).

Havelock (1973) described innovation as a change process. Planned change or planned innovation was "change or innovation which came about through a deliberate process which is intended to make both acceptance by and benefit to the people who are changed more likely" (p. 5). He identified that educators at all levels of a school system were involved are identified them as change agents--a person, group, organization or community who facilitates a change or planned innovation.

Fullan (1982) was concerned with change in educational environments. He found that there were claims that some schools were being bombarded with change while others were claiming that there was nothing new under the sun. He observed that there was often confusion between the terms change and progress (p. 4) and that resisting some changes might be more progressive. "One of the most fundamental problems in education today is that people do not have a clear, coherent sense of *meaning* about what educational change is for, what it is, and how it proceeds" (p. 4). Fullan stated that the problem of meaning was central to making sense of educational change. In order to achieve greater meaning, both the small picture (subjective meaning or lack of meaning for individuals) and the big picture (the sociopolitical process) must be understood. In order to examine both of these pictures, it was necessary to contend with both the *what* of change and the *how* of change (p. 4).

Fullan described researchers as generally seeing three broad phases to the change process. Phase I was described as initiation, mobilization, or adoption. It consisted of the process leading up to and including a decision to adopt or proceed with change. Phase II was the implementation or initial use and involved the first experiences of attempting to put something into practice. Phase III dealt with continuation, incorporation, or routinization (Figure 7). To these three phases Fullan added a fourth--outcome.

This figure has been removed because of copyright restrictions.
Figure 7
Change Model (Fullan, 1982, p. 40)

In Fullan's model *"change is a process, not an event"* (p. 41). What happened in one stage strongly affected subsequent stages while new determinants emerged in the subsequent stages. Fullan cautioned that the process did not occur in one direction only (the two-way arrows) and that events at one phase could feed back to alter decisions taken in previous stages. Other factors not included in the model but described by Fullan were the scope of the change and who initiated the change as well as the factor of total time and the inability to precisely demarcate the sub phases.

Initiation

Examining the factors influencing whether or not changes get adopted provides an understanding of the course of changes. Edwards (1987, p. 21) cited the work of Berman (1981) that identified five categories of factors affecting the educational change process: local contextual conditions, primary attributes of change efforts, local policy choices, endogenous variables, and external factors. Fullan (1982) identified the factors that influenced the change process and related those factors to each of the phases of the change process. The ten factors associated with adoption of changes in educational settings were:

1. Existence and quality of innovations
2. Access to information
3. Advocacy from central administrators
4. Teacher pressure/support
5. Consultants and change agents
6. Community pressure/support/apathy/opposition
7. Availability of federal or other funds
8. New central legislation or policy (federal/state/provincial)
9. Problem-solving incentives for adoption
10. Bureaucratic incentives for adoption (Fullan, 1982, p. 42).

The absence or presence of these factors influenced decisions to either reject or adopt specific change programs, directions, or policies. The combinations among the factors were also important, citing the example of community pressure or support combined with a problem-solving orientation as having different consequences than when it was combined with a bureaucratic orientation.

Fullan's ten factors were not the only things that had impact on the adoption of innovations. The *existence and quality of the innovations* were important to adoption. Fullan found that there were many educational innovations in existence. They were usually developed in response to an incentive system of society (e.g., market conditions such as government sponsorship, publishers for a national market, etc.). While noting that the quality of programs was difficult to assess and agree upon, the quality of the new programs and the clarity of the programs were often difficult problems to solve.

Fullan also found that differential *access to information* accounted for the differences in the adoption process. School district personnel often spent more time attending conferences and workshops than did teachers. Hence teachers were less likely to come into contact with new ideas. Access to innovations were also dependent upon an infrastructure of communication (e.g., ease of transportation, resources, density of population, etc.) that resulted in urban school districts enjoying more favorable conditions than rural and smaller jurisdictions.

"Educational adoption never occurs without an advocate" (p. 45), potentially one of the most powerful being the chief district administrator along with his/her staff. *Advocacy* and initiation of change was key to successful adoption of innovative programs. This illustrated the impact of central administrators who could not only sponsor but could also block change programs.

While most teachers innovate, teachers as a group do not have the same level of opportunity to come into contact with new ideas nor the time and energy to follow through on them. *Teacher advocacy* is necessary for the adoption of change at the individual classroom level, particularly when the innovation is clear and practical and district personnel support the innovation.

Adoption was also impacted by the presence of *linking agents* or consultants (district support staff, provincial agents, etc.). Linking agents help teachers adopt innovations that teachers wanted. How closely the innovations were related to the needs of teachers and others in the school district depended on whether the funding was oriented to bureaucratic grants or to the problem-solving process (p. 47).

Communities vary and characteristics of school districts differ. Different combinations of these two variables make it difficult to identify why some communities support innovation, why others block them, and why others are apathetic. *Community pressure, support, opposition, or apathy* can put pressure on administrators to "do something" or to oppose potential adoptions that they are aware of. Citing the work of Berman and McLaughlin (1979), Fullan (p. 47) noted that major demographic changes such as rapid growth or change in composition lead to the development of community efforts and demands for change.

Other variables found to be significant in the adoption process were: the *availability of government or other funds* and *new central legislation or policy*. The availability of funds showed "beyond much doubt that the availability of resources external to the district is a powerful stimulant for adoption" (p. 49). Legislation often mandated adoption at the local district level. Without the existence of lobby groups, new social change programs would likely not get formally adopted. General policies were often left ambiguous with local district levels to adopt.

Problem-solving orientations and *bureaucratic orientations* guided and drove many of the other factors. Fullan noted the existence of districts that were problem-oriented. These districts identified local needs, were dissatisfied with existing performance, and sought out resources. Opportunistic or bureaucratic districts were influenced largely by three factors: bureaucratic safety (when innovations added resources without requiring behavioral change), response to external pressure ("adoption" eased pressure), and approval of peer elites (whatever is popular among leading professional peers). "Bureaucratically speaking, the political and symbolic value of adoption for schools is often of greater significance than the educational merit and the time and cost necessary for implementation follow-through" (p. 51).

Implementation

Fullan (pp. 56ff) identified 15 factors affecting implementation. These factors were divided into four main categories (A - D). Category A referred to attributes of the change itself. Categories B and C consisted of characteristics of the district and of the school as a unit. Category D was concerned with factors external to the school system. An elaboration of the four categories is listed below:

A. Characteristics of the Change

1. Need and relevance of the change.
2. Clarity
3. Complexity
4. Quality and practicality of program (materials etc.)

B. Characteristics at the School District Level

5. The history of innovative attempts
6. The adoption process
7. Central administrative support and involvement
8. Staff development (in-service) and participation
9. Time-line and information system (evaluation)
10. Board and community characteristics

C. Characteristics at the School Level

11. The principal
12. Teacher-teacher relations
13. Teacher characteristics and orientations

D. Characteristics External to the Local System

14. Role of government
15. External assistance (Fullan, 1982, p. 56).

Fullan cautioned that the fifteen factors causally influenced implementation in the direction of some sought after change. If any one or more factors were working against an implementation, then the process would be less effective. The factors were also to be thought of as "*a system of variables that interact*" (p. 57) and were not in isolation from each other.

Continuation

Fullan noted that one of the most discouraging prospects in understanding the implementation and continuation process was the realization "that it is not linear and is never-ending" (p. 77). The factors could not be dealt with one at a time but had to be continually borne in mind and attended to when the need arose. The single most powerful internal factor influencing continued change was staff and administrative turnover. Since effective change involves interaction among users, the removal of key users weakens the conditions that would incorporate or generate new members.

Innovations are supposed to accomplish something worthwhile, both in the short-term and in the long-term. Five different kinds of outcomes were identified, generally from the short to long-term:

1. Degree of implementation
2. Attitude toward innovation
3. Impact (a) students' benefits
   (b) teachers' benefits
   (c) organizational benefits
4. Continuation or institutionalization
5. Attitude toward school improvement (p. 77).

The degree of implementation assessed the degree of actual change. Attitude toward innovation was concerned with the perceptions of the strengths and weaknesses of the change. Impact involved an assessment of student learning, teacher benefits, organizational benefits, etc. Continuation involved incorporation into the budget, staffing, etc. Attitude toward school improvement was

> a kind of meta-variable related to whether the experience with the change effort increases or decreases people's attitude toward engaging in new school improvement programs--in brief, whether the experience has led people to conclude generally that is worthwhile to try and implement changes (p. 78).

Boundary Spanner

Tushman (1977) described the need for an innovating system to gather information from and transmit information to external information areas. He found that special boundary roles evolved in organizations' communications networks to fulfill the linking between the organization's internal network and its internal sources of information. These boundary roles were found to be important in the process of innovation. Citing the work of Myers and Marquis (p. 588), Tushman described the innovation process and its key communication domains. Innovation occurred in three phases: idea generation, problem solving, and implementation. During the idea generation phase new ideas or new approaches could be found most effectively outside of the innovating organization. During the problem solving phase there had to be a substantial exchange of information within the developing unit and its external areas. During the implementation phase there had to be effective coordination among all functional areas. "While the need for communication and information will vary over time, this approach suggests that to be effective, the innovating sub-unit must efficiently gather information from and transmit information to several external information domains" (pp. 588-589).

Tushman found that one way organizations dealt with communications across organizational boundaries was to develop individuals "capable of translating contrasting coding schemes and therefore acting as boundary spanners between the work unit and external information areas" (p. 591). He suggested that these *gatekeepers* did not attend to all external communication areas but only to those critical to the work of the sub-unit (p. 592). Tushman's research indicated that

> For high performing projects, the number of boundary roles per project is contingent on the nature of the project's work. The more the uncertainty in the task, the greater the information-processing requirements and, up to a point, more roles evolve to deal with that increased uncertainty. Thus, beyond the cross-boundary communication function, boundary roles also seem to be an effective way of dealing with extra-unit uncertainty (p. 601).

Tushman pointed out that one of the implications for managers of his research was that those interested in managing innovation should explicitly recognize the importance of key individuals in the system's communication network.

> Not only are special boundary roles an efficient mechanism for gathering and transmitting information, but evidence also suggests that they are important in other organizational areas (for example, as integrators or as implementors of management information systems), given their exposure to critical information and their informal status (pp. 602, 603).

Havelock (1973) noted that planned innovation has been analyzed from a number of viewpoints. He found "that most practicing change agents organize their work and their thinking about innovation in terms of specific projects in which they are involved, projects that have a defined beginning and an end, and a sequential history" (p. 5). A change agent acted in four primary ways: as a catalyst, as a solution giver, as a process helper, and as a resource linker. The catalyst functioned to overcome the inertia preventing change. There are those who have definite ideas about problems and what the change should be. These function as the solution givers. The agent acting as process helper helped the client recognize and define needs, diagnose problems and set objectives, acquired resources, selected or created solutions, and so on. The agent that acted as resource linker brought together needs and resources (financial, knowledge of solutions, diagnostic skills, etc.).

Levin (1981) described the leadership role of change agents. Leadership came from "leaders" and "fixers." "Leaders" motivated change and gained support for an innovation through individual characteristics and personality. "Fixers" were task oriented individuals who related to others in some instrumental way during the implementation process.

Morrish (1976) described some of the specific qualities of innov: ors. Innovators were generally young, had high social status, possessed interpersona: nd cosmopolite sources of information, traveled and related outside their system, exercised opinion leadership, and were viewed as deviants by both their peers and themselves. Balistreri (1987) described general qualities of a change agent as:

> 1. The vision to see beyond the present
> 2. Capable of conceptualizing needed changes.
> 3. Dedicated to an idea.
> 4. Energy and persistence to pursue an idea.
> 5. Knowledgeable about the issues, elements and factors associated with the proposed change.
> 6. Analytical and objective in his/her thinking.
> 7. Positive attitude toward change as well as the potentials for change (pp. 3-5).

Summary

Case studies have been written on the implementation of computer systems in a variety of settings. Kiesler and Sproull (1987) collected fourteen papers outlining the acquisition and implementation of computers at Carnegie Mellon during the 1980s. Long (1988) wrote a case study for those aspiring "to bridge the gap between computer/MIS concepts and business practice" (p. xi). While dealing primarily with management information systems (MIS), many of the other administrative computing areas identified in Long's study were: organization, integrated data base, word processing, electronic mail, voice mail, microcomputer proliferation, etc. These are technical areas that are important to computer implementation. These areas are similar to areas discussed by Bluhm (1987), Martin (1984), and others.

The change process, as outlined by Fullan (1982) and others, deals with organizational issues in the design and implementation of change. Technical computing literature often avoids these areas. Mintzberg (1989) discussed particular components of organizations and identified school jurisdictions as professional organizations. Any attempt to successfully design and implement administrative computing must involve a careful examination of the organizational structure both at the macro level as well as within the developmental groups.

Hersey and Blanchard (1988) demonstrated that there were different management styles that managers could utilize. These styles depended upon the readiness of the staff to perform the tasks and the type of tasks involved. Both senior administrators and developmental groups need to diagnose the situation and alter their leadership styles accordingly. In addition to different styles, Hersey and Blanchard noted the necessity to consider time as a factor and to develop the long-term view of staff development. All of these aspects, therefore, are necessary for an understanding of what is necessary for the successful design and implementation of a computerized student information system in a school district.

Chapter 3

Methodology

Case studies are often used to "explore the processes and dynamics of practice" (Merriam, 1988, p. xi). Merriam noted that case studies have offered insights into educational practice and proved to be helpful in forming policy. Merriam (1988, pp. 11-13) noted that there were four characteristics that were essential properties of a qualitative case study: particularistic, descriptive, heuristic, and inductive.

The *particularistic* characteristic refers to case studies that focus on a particular situation, event, program, or phenomenon. The case itself is important for what it reveals about the phenomenon and for what it might represent. Merriam (1988) noted that the specificity of focus of particularistic case studies makes it an especially good design for practical problems--for questions, situations, or puzzling occurrences arising from everyday practice. Case studies "concentrate attention on the way particular groups of people confront specific problems, taking a holistic view of the situation" (Merriam, 1988, p. 11).

Merriam (1988) also noted that there were three statements that reflect a case study's particularistic nature:

- it can suggest to the reader what to do or what not to do in a similar situation.
- it can examine a specific instance but illuminate a general problem.
- it may or may not be influenced by the author's bias (Merriam, 1988, p. 13).

It was the ability to "suggest to the reader what to do or not do" in an educational computing situation as well as "specific instances that illuminate a general problem" of decision support system development in education that interested the author.

Inductive means that, for the most part, case studies rely on inductive reasoning. Generalizations, concepts, or hypotheses emerge from an examination of data--data grounded in the context itself. Occasionally, one may have tentative working hypotheses at the outset of a case study, but these expectations are subject to reformulation as the study proceeds. Discovery of new relationships, concepts, and understanding, rather than verification or predetermined hypotheses, characterizes qualitative case studies.

Merriam (1988) suggested that the *heuristic* quality of a case study is suggested by the following aspects:

- it can explain the reasons for a problem, the background of a situation, what happened, and why.
- it can explain why an innovation worked or failed to work.
- it can discuss and evaluate alternatives not chosen.
- it can evaluate, summarize, and conclude, thus increasing its potential applicability (Merriam, 1988, p. 14).

A case study utilizing a blend of the particularistic and heuristic properties was conducted.

Conceptual Design

The conceptual basis for this study is based upon models derived from Fullan's (1982) change model, Mintzberg's (1989) organizational model, Martin's (1984) prototype design model of user driven systems, and Tuschman's (1977) concept of the boundary spanner. Aspects of each of these models demonstrate the multifaceted nature of this case study.

User-driven systems, according to Martin, include:

- user-seductive software
- well human-factored computer screens and interactive devices
- self-teaching software and HELP functions that encourage the user to experiment...
- linkage to on-line data systems and data-base support. ..."(Martin, 1984, p. 52).

Martin (1984) proposed a model for "information engineering."

> The term information engineering refers to the set of interrelated disciplines which are needed to build a computerized enterprise *based on data systems.* The primary focus of information engineering is on the data that are stored and maintained by computers and the information that is distilled from these data (Martin, p. 92).

Martin maintained that the basic premise of information engineering is that data lie at the centre of data processing (Figure 8).

This figure has been removed because of copyright restrictions.

Figure 8
Information Engineering (Martin, 1984, p. 93)

Martin cited two basic premises of information engineering: data lie in the centre of data processing and types of data used in an enterprise do not change very much (the values of data change, but the structure and attributes do not). The processes on the left of Figure 8 *create* and *modify* the data and the processes on the right of Figure 8 *use* the data.

The identification of the data requirements involves the participation of all stakeholder groups as participants. The attributes of the data thus identified represents the core of the requirements for the student information system. The values of the data change constantly, and using the data in a consistent user friendly interface improves the likelihood that the data will actually be utilized by the various stakeholder groups.

An Integrated Model

Traditional approaches to computing, as noted by Martin (1984), resulted in a crisis of application development in computing operations. To meet this crisis Martin articulated a user driven computer model. While Martin defined this user driven model, he did not describe the process of implementing the change that occurred in the creation of prototypes or in their implementation. Fullan (1982) dealt with implementing change in educational environments. He described three broad phases for change in education. These phases are not well addressed in the information engineering literature (Martin, 1989), but constitute a very integral part of the development of computing applications in an educational environment. The two domains of the educational administrator on the one hand and the computer development group on the other are very distinct. The role of the boundary spanner was utilized to bridge the gap between these two groups. These models are integrated into a composite model (Figure 9).

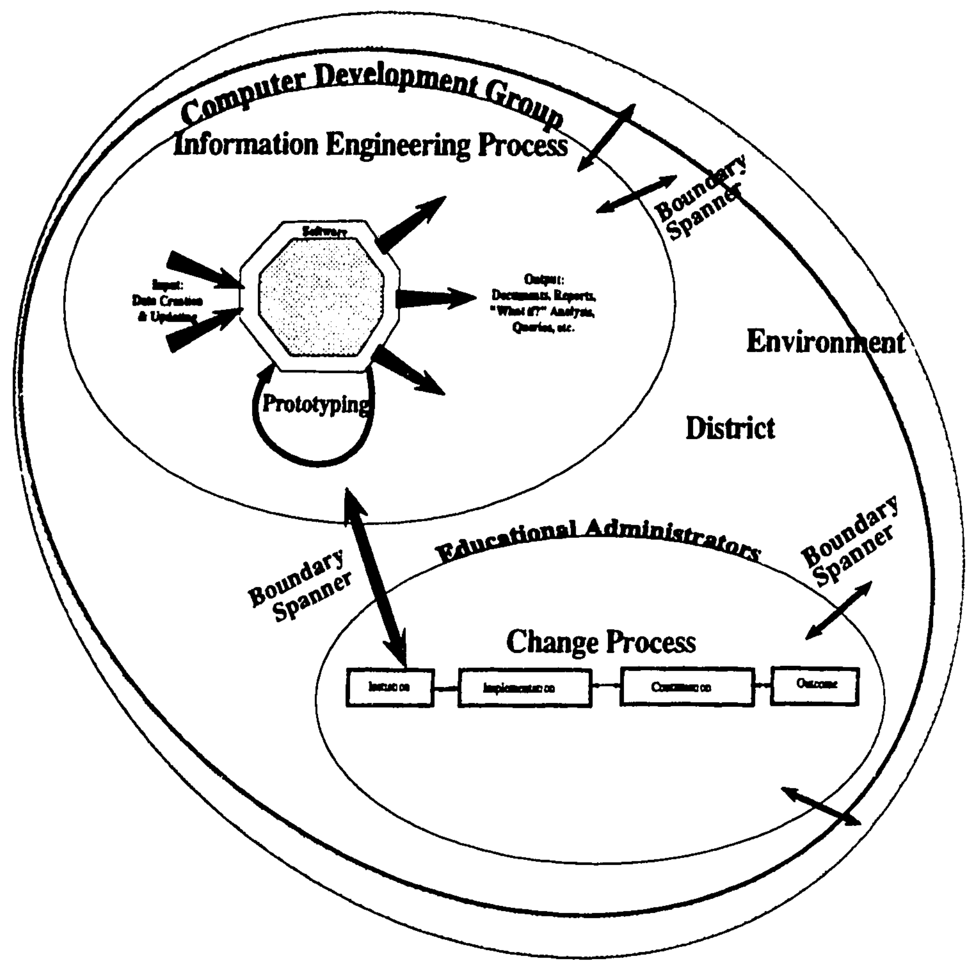

Figure 9
Integrated Model

Limitations

Merriam (1988) noted that "In a qualitative case study, the investigator is the primary instrument for gathering and analyzing data" (Merriam, p. 36). In this study, the author has been heavily involved in the project under review and must make every effort to minimize the subjective biases associated with being the primary researcher. "Facts cannot be assumed to be independent of the inquirer's values or of the theoretical language he or she brings to bear" (Lincoln & Guba, 1985, p. 332). One of the potential benefits of "participant as researcher" in a case study is the insight that can be brought forward. This must be balanced against potential biases introduced by actually participating in the study.

Delimitations

The study was restricted to a case study of the development of the SIS in County X. It dealt only with administrative computing in the development of the SIS. It did not examine the use of computers in instructional areas and only examined those aspects of administrative microcomputing which impact the SIS (primarily architecture of the telecommunication infrastructure). The duration of the study was from September 1988 through to the end of the school year in August 1989.

The scope and complexity of the SIS was outlined (Figure 10). It was composed of a number of major interrelated subsystems. This study only dealt with the final system design and did not involve other designs that were examined and discarded. Digression to these other designs only occurred where it was thought to be of significance to educational administrators.

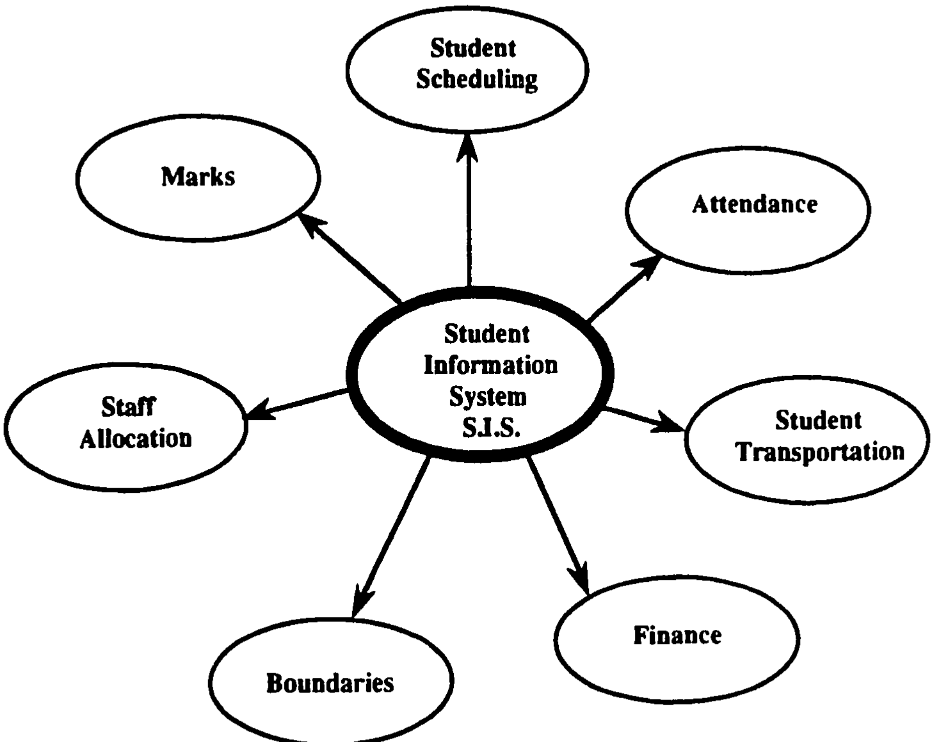

Figure 10
Major Components of the SIS System

Not all of the material collected in a case study would be of importance to the study findings. March (1987) wrote that

> A writer who speculates about the future of social institutions or technology faces a dilemma. On the one hand, we know that most interesting speculations about the future involve surprises, things that are forecasted to change our lives dramatically and unexpectedly. On the other hand, we know that most of the surprising things we might predict about the future are almost certainly wrong. Some surprising things will certainly happen next year, but the likelihood that any particular surprising thing will happen is close to zero (March, 1987, p. 18).

The study dealt only with those aspects of potential importance to the development and implementation of decision support systems and did not deal with speculation/predictions of the future.

The computer source code for the programs was not examined. The computer programming language chosen for developing the SIS system was examined. Those aspects of the language that effect the outcome of the system or assist in demonstrating a solution to an education administration problem were examined.

The provision of training (signing-on to the system, running programs, creating reports, use of various packages, and so on) was an important aspect of any implementation. Training tends to transcend many facets of a development cycle. This area was not addressed.

Definition of Terms

In the thesis, there are a number of technical terms used which have meaning in data processing/computing environments (refer to Appendix A for a glossary of terms). Many of the terms are defined in the body of the thesis in order to aid in the general readability of the thesis.

Data Collection

The data for this thesis came from the following sources: Superintendent's Team Minutes, Board of Education Minutes, SIS Steering Committee Minutes, SIS project reports, SIS Handbook, and interviews with key participants. The majority of the information came from two sources: the project meeting reports, which were extensive and were produced biweekly, and the Steering Committee minutes, which were the "semi official" documents on the project. Information on the attributes of the data base files was derived from the data definitions of the finalized data structures. As in any prototyping environment, data bases definitions changed as the project proceeded

Summary

Fullan (1982) noted that "...our cognitive ability to *conceptualize, understand, and plan the social processes* of educational change represents the most comprehensive and generative resource for dealing with change" (p. 93). He noted that such knowledge, once obtained, was a far more powerful resource than a memorized list of steps to be followed. Computerized systems are an area of education administration which is not well understood or researched. It was my desire to demonstrate, via this case study, many of the change processes which can be successfully applied to a computer systems development in an educational environment.

Chapter 4

The Case

This case occurred in a county (County X) within the Province of Alberta. County X was one of the largest school systems in the province at that time. There are two components to a county system--education and municipal. The municipal component deals with areas of civic government and has county councilors who are elected for three year periods. The areas of civic responsibility include taxation, roads, recreation and parks, development, police, fire, and so on. The education component deals with areas of education such as instruction, student transportation, and so on.

In county systems, councilors are also trustees on the school board. Councilors fulfill their municipal functions in county meetings and their educational functions in board of education meetings. The municipal administrator deals with municipal matters and the superintendent deals with educational matters. Under Alberta legislation, county education budgets must be submitted to a municipal county meeting for money bylaws and for the setting of property mill rates. This is due to the fact that the legal basis for money bylaws and mill rates reside with the municipality.

County X had been reorganized internally many times during the previous 15 years in an attempt to improve its operation. County X contained a large urban population as well as a large number of "county residential" (acreage developments) and traditional rural properties. Due to the large urban population in the county studied, the provincial Municipal Act was modified to allow for three additional "trustees at large" from the urban population in addition to the councilors representing the urban area. Three trustees at large were also elected from an autonomous urban area that was part of the school system but was governed by a different municipal government. There were sixteen trustees--ten councilors and six trustees at large (Figure 11).

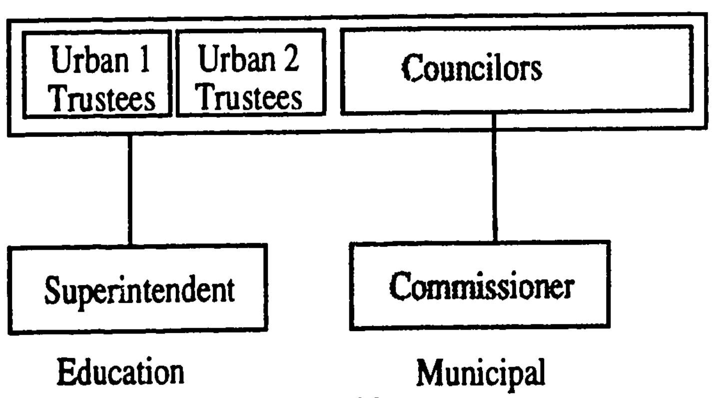

Figure 11
County Form of Governance

In 1982, County X lost a large portion of its geographic area through annexation by an adjacent municipality. In addition to the area lost, 90% of its light industrial assessment and 45% of its heavy industrial assessment were also lost. In the aftermath of these events, the Commissioner of the municipal component left the County. At that time both the County Council and Board of Education decided that there was a need to improve relationships between the municipal and education components of the organization. In an attempt to harmonize the educational and municipal components, an acting Commissioner was selected and charged with creating a new corporate structure. One of the results of this new corporate structure was the creation of a Corporate Services Department. This department, headed by the Treasurer, was to provide "shared" services (finance, computing, purchasing, etc.) to the entire county (Figure 12).

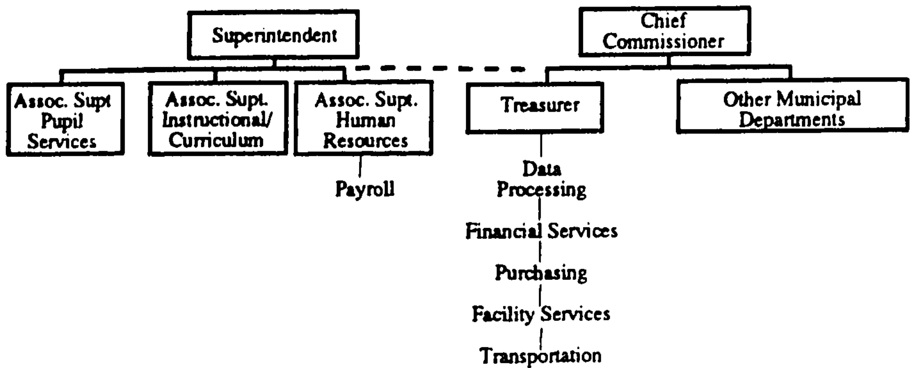

Figure 12
Corporate Services Structure

At that time, the separate education and municipal human resource departments were combined under the supervision of an Associate Superintendent and the payroll function was moved from the Finance Department into the newly created Human Resources Department. Sub-organizational structure was later modified to reduce the work load of the Treasurer.

SIS

The SIS project was a large project that took place over a number of years. The major emphasis of this study was the period from September 1988 to August 1989. The antecedent events in the development of the SIS were referred to as "Phase I" by the project team. This included the first versions of the software written in COBOL. The study period was referred to as "Phase II." Events after the study period were referred to as "Phase III". The project schedule chart (Figure 13) was the working chart which outlined the tasks and the sequencing of events. These tasks were divided into two parts: Part A began with the high level design of the system and Part B began with the milestone event of the schools being "on-line" for advanced registration and ended with the completion of training and system testing.

FIGURE 13: SIS PHASE - 11 SCHEDULE CHART

REMOVED DUE TO POOR PRINT QUALITY

The schedule chart is a PERT (Program Evaluation Review Technique) chart. Each element (task) in the schedule chart is defined on the legend (lower right corner, Figure 13). Each task presents key information for the task: the earliest start date, the latest finish date, the task, and the number of programmer days. The "earliest start" date and "earliest finish" date (format is year/month/day) appear at the top left of each box while the length of each task appears at the top right of each box. The first task, "High Level System Design" was scheduled to begin September 1 1988 (88/9/1) and to finish no later than September 12 1988 (88/9/12). The length of time allocated to this task was eight days.

The length of time for a task was referred to as the number of "Nathan days." This term, named after one of the participating programmers (used by permission), was coined to determine the length of time that a particular task would take to complete. Estimates of the number of days of programmer time are extremely difficult to determine since various programmers, some of whom were contracted from outside the organization, were working on the project. A common measurement for the length of time to complete a task was required. This is similar to the use of constant dollars in financial projections.

A "Nathan day" was defined as the amount of work that could be performed in one day by a senior programmer who was familiar with (a) computerized student records systems, (b) the operating system, (c) the programming languages involved, and (d) the overview of the project. If a programmer was unfamiliar with one or more of these areas of expertise, then various factors were applied to the time estimates in order to determine the actual number of days that should be allocated to the task. Outside programmers were initially assigned a rating of 1.5 times the number of days estimated in the "Nathan days" projected for the task. A given task, such as the "Report Interface Program", was estimated to take 20 "Nathan days". This same task would be estimated to require 30 days from an outside programmer that was fluent in the programming language and operating system, but not in the project itself. Project estimation proved to be a difficult administrative problem since it was imprecise and altered the estimated costs of the project.

SIS Assumptions

Senior administrators in County X were very interested in answers to questions typified by "How much will it cost?", "When will it be done?", "What are the implications for the district?", and so on. The schedule chart was a useful tool in communicating the *sequencing* of tasks and in the *identification* of key events, costs, and manpower requirements for senior administration and other stakeholders. Communicating highly technical events, particularly some of the difficulties and various trade-offs that are a constant part of a computerization project required for administrative direction, was a difficult task.

The key administrative group within the education administration was the Superintendent's Team. This team consisted of the Superintendent, three Associate Superintendents, the Board Secretary, and the County Treasurer. This team met every Monday morning for one-half day and discussed issues and directions for the district. Reports outlining alternatives, strengths and weaknesses of the alternatives, and recommendations to which alternative was recommended were regularly received by this Team. Regular status reports on the SIS project were brought to the Superintendent's Team. One report brought forward to provide background on completion dates and therefore "probability of success" outlined some of the *assumptions* underlying the schedule chart (Figure 14). These assumptions were

SIS Rewrite: Phase II -Assumptions

At a meeting of the project leader with the Corporate Computer Department (July 14, 1988) the assumptions behind the schedule tasks were discussed. In summary they are:

Option 1: Target date - March '89/Sept. '89

1. Estimates of time are correct.
2. High calibre of contract programmers.
3. The task list identifies all tasks and is complete.
4. Little or no slack time exists in chart and any overruns are serious.
5. No staff turnover will occur during the project.
6. No one is absent due to sickness or other reasons.
7. Staff are not reallocated to other emergency tasks and removed from the project.
8. No time is given to major testing; there is no pilot project in a school.
9. The team members are "fresh" (no burnout).
10. There will be no changes to: report cards, scheduler, reports, registration, etc.
11. Secondment can complete the job in first semester.
12. There are no major undetected problems in Phase I.
13. Contractors can perform the "farmable" tasks.
14. Multiple tasks can be performed concurrently by scarce programmer staff.
15. There is no contingency plan.
16. Other potential valid options like scanners are not examined.
17. No changes in the number of schools using attendance/report cards during project.

Probability: Part A ~70%, Part B < Part I

Option 2: Target Date - December 1989 milestone

Gain:

1. Increased slack time.
2. Allow for some movement on quality of contractors.
3. Allow limited changes.
4. Gain a second budget year.
5. Potential to give more tasks to contractors.

Lose:

1. No cut over mid year
2. Scheduling March 1989 difficult - run on old system for Sept. 1989.
3. New bridges needed old system-->new system that are not on timeline. Would add ~30 "Nathan" days to project.
4. Midyear testing non-existent.
5. Principals' letter is on the table.
6. Potential for Project Manager absent Sept. 1990.

Probability: ~90%.

Option 3: March 1990 with old system kept until July 1990 (not proposed or supported).

Gain:

1. Eliminate some/all contractors to existing staff.
2. Eliminates "fudge" factors.
3. Possible to add more people (Internal Letter, July 14, 1988)

Figure 14
Schedule Chart Assumptions

The administrative assumptions underlying the project tasks illustrated the difficulty the project team had in estimating time and, therefore, the costs of a major computer project. Three options were presented to the Superintendent's Team. In the first option, seventeen assumptions were noted. These assumptions included the following: no staff turnover (assumption 5), no absences (assumption 6), no relocation to other high priority tasks (assumption 7), no functional changes to the existing system (assumption 10), and so on. Given that all seventeen of these assumptions were valid, the probability of the project being completed on time was set at 70% for Part A and a lower probability targeted for Part B. The project timelines were ambitious and held an element of risk in the successful on-time completion of the tasks. Due to the small size of the computing group, the amount of effort in keeping accurate time tracking, and the short timelines for the project, a decision was made not to estimate or monitor time for each project element. Time was tracked in *general terms* to ensure the project was basically on schedule.

Option 2 was presented to demonstrate the "trade offs" of moving the project target date to December 1989. Option 3 was presented for the sake of completeness of the various alternatives but was not recommended.

Key Events Timelines

Many events were involved in the development and implementation of the SIS. The integrated model (Figure 9) illustrated an environment as a backdrop to the events in the organization. This environment consisted of historic events, organizational events, and external events that had an impact on the case. Since a case study is "frozen" in time but exists within a context of antecedent events, a chronology of the events leading up to and during the case provides a context for the study. These events are broken into *antecedent* events; those events and/or decisions that had an impact on the time window selected for the study, the *study* events; those events/decisions that occurred during the study window, and *emergent* events; those events/decisions that emerged after the study or are likely to emerge (the emergent items appear in the author's comments).

Antecedent Events

The event which was arbitrarily set to be the start of this study was the decision to evaluate various available microcomputer software packages (September 1988). A decision was made to determine whether a commercial package was available to solve the conversion problem or if the existing student information software had to be rewritten. The events prior to September 1988 (Figure 15) provide the *context* for the events that occurred during the study. The antecedent events outlined in Figure 15 provide a *context* for the three major threads of organization, computing events, and computing industry by outlining major events in each of the three areas. These events were separated into three major threads: organizational events, district computing events, and computing industry events.

"Organizational Events" deal with such things as structure, reorganization, personnel turnover, and so on. "District Computing Events" on the other hand deal with hardware, software, computing languages, and so on, that took place within County X. "Computing Industry Events" deal with hardware, software and other events in the computing environment outside of the County.

The integrated model (Figure 9) indicated events within County X's domain in two major spheres--the computer development group and the educational administrators (primarily the Superintendent's Team). Each of these groups possessed its own culture and interacted with each other, with other departments and schools in County X, as well as with the larger environment outside County X. Three figures (Figures 15 "Antecedent Events", 17 "Case Events", and 26 "Emergent Events") serve as a chronology of events for the study.

Antecedent Events

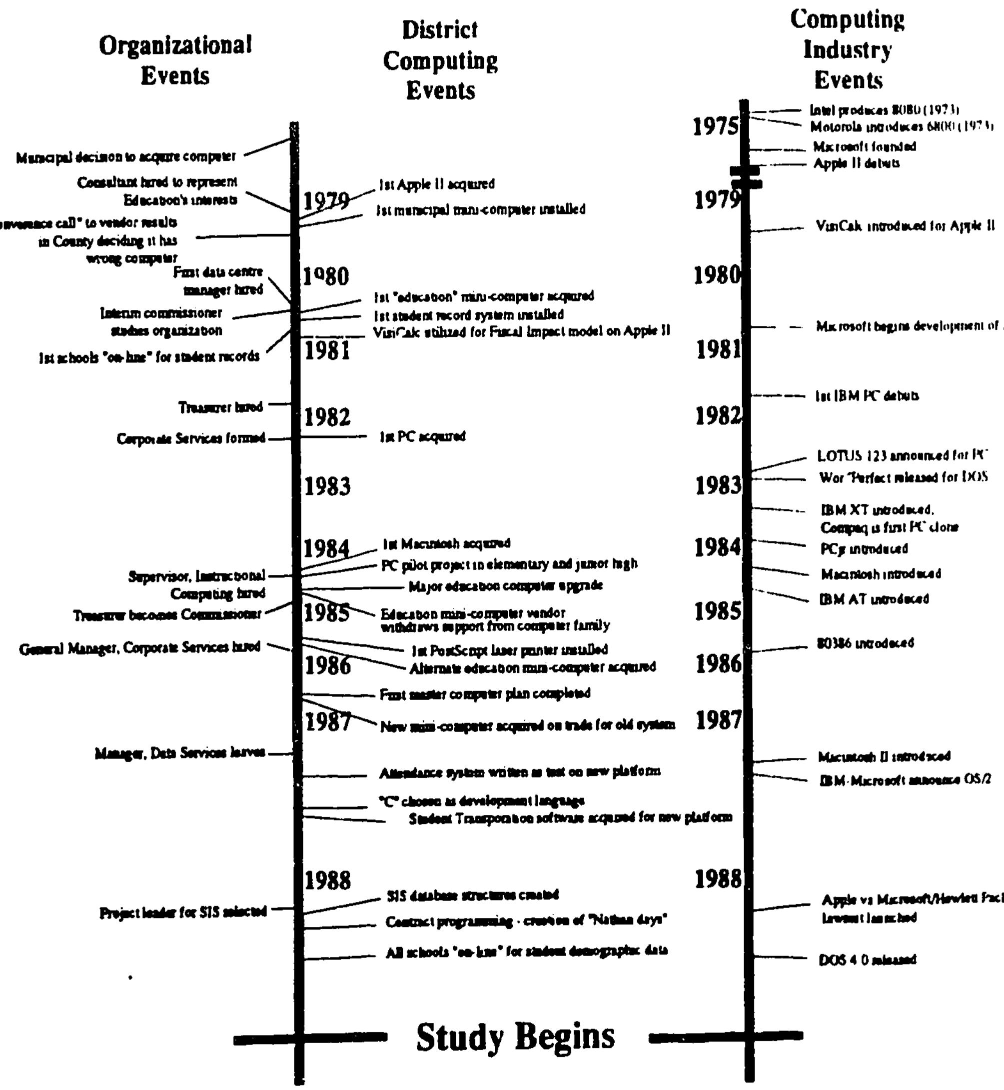

Figure 15
Antecedent Events (prior to 1988)

Antecedent Organizational Events

Examination of the first two organizational events, the "municipal decision to acquire a computer" and "a consultant hired to represent Education's interests" illustrated the "municipal versus education" *climate* of the County early in the development of computing. A computer was acquired for financial applications. *Municipal* administrators intended that this corporate computer be used by both education and municipal finance for their financial record keeping. *Education* administrators had a concern that a financial computing system would not be able to perform other more educationally oriented tasks such as student scheduling, report cards, and so on. A consultant was hired to evaluate the municipal computer and make recommendations to the Board of Education on the computer's functionality.

Events in County X continued to evolve and change. In November, 1980, a new interim municipal commissioner was hired. One of the commissioner's mandates was to devise an organizational structure to permit "economies of scale" by sharing of services required by both municipal and education components of the county. A Corporate Services Department which was responsible for computing, finance, transportation, etc. was formed in May 1982. The General Manager of this department also served in the capacity of Treasurer for both the Council and Board. This resulted in the computing department reverting to the control of a financial person instead of an educator.

The General Manager/Treasurer had a strong commitment to make the "spirit and letter" of the new arrangement work. This person remained in that position until September, 1985, when a further reorganization took place with the promotion of the Treasurer to Commissioner. At that time a search was made to replace the treasurer's position and advertisements were published.

Since it was believed that the work load of the corporate services manager/treasurer was too large for one person, responsibility for the Facility Services Department was removed from Corporate Services and placed under the control of one of the Associate Superintendents. At the same time, responsibility for the supervision of the Transportation Department (which included both public and student transportation) was assigned to the same Associate Superintendent. The Superintendent and Commissioner selected different candidates for the position of Treasurer. Since there was not an agreement on a joint candidate, an advertisement for Treasurer was undertaken a second time to find a candidate who was mutually acceptable. After a second series of interviews, a new Treasurer was selected.

These events illustrated the divided nature of a county form of governance. Even though there was a clear political desire for economies of scale by sharing corporate departments between Municipal and Education wherever possible, the structure of the elected officials (Figure 11) remained in multiple camps--municipal councilors and trustees from two separate urban areas. The differing priorities of two separate organizations and two different management approaches resulted in the difficulty in selecting a suitable Treasurer. This had an impact on the subsequent events in the development of all computerized systems.

A new Treasurer was hired in September 1986. This Treasurer had a different outlook on computing issues than the Computing Manager. For example, the Treasurer favored the purchase of software packages instead of the maintenance of the existing systems that were developed within the County. The established organizational computing climate was that programmers were professionals and as such were given a great deal of professional latitude in determining solutions to computing problems. With the new

Treasurer came a philosophy that programmers "did as they were told" and that "Managers make all decisions." The first meeting of the Treasurer with the computing group was opened by the statement, "It's kick ass time" (personal interview, February 1987). The manager of the Data Centre resigned from the County in May 1987. This departure increased the climate of uncertainty within the programming staff. Another result of the manager's departure was the affirmation that computer software development would be conducted with the traditional Carlsen & Lewis style model.

Antecedent District Computing Events

The "state of the art" in commercial computing in the late 1970s was oriented towards mini-computer and mainframe computers. The Apple II, the first microcomputer widely acquired by school districts in Alberta, had only recently been invented. County X acquired its first Apple II in early 1979. This began a trend by the education component of County X to acquire new computer technology shortly after it became available. This trend continued throughout the case study and is illustrated by other technology acquisitions.

A municipal computer was installed in early 1979, but was determined to be inadequate for the educational needs of County X. A conference call (March 1979) placed jointly by education and municipal representatives to the computer manufacturer verified that the computer was inappropriate for the educational needs of the district. The manufacturer agreed that there were no software packages available for student records, scheduling, transportation, and so on for that model computer. It was further indicated by the manufacturer that there was a five year lease agreement in place and any attempt to break the contract would result in litigation.

After it was determined that the existing municipal computer would not meet the student information needs of the Board of Education, the Board established its own computing group. The Board purchased its own computer system and a Data Centre Manager, the first computer professional in the County, was hired in May 1980 to head an education computing centre. *Subsequent to* the hiring of the manager but *prior to* acquiring the first education computer, County Council decided to consolidate all computing within the education group headed by this manager. The manager was, therefore, responsible for the existing municipal computer as well as the development of a new computing environment.

One of the mandates required of the manager was the implementation of a pupil record system. The Board issued a request for quotation (RFQ) for its computer system. Part of the RFQ required the vendor to supply a student record package, along with the computer equipment. This pupil record system was a mandatory requirement in the purchase agreement. The successful vendor supplied an on-line, interactive student record system which was implemented in 1980.

Antecedent Computing Industry Events

The microcomputer revolution began with Intel's introduction of the 8080 CPU in 1974. The largest software company in the world, Microsoft, was founded in April of 1975. The Apple II microcomputer was introduced in April of 1977. This computer made widespread impact on most Alberta jurisdictions with the Apple IIe and Apple IIgs still widely used in many districts (1992).

VisiCalc, the first spreadsheet program for microcomputers, was introduced in June of 1979. Introduced on the Apple II, this software package helped create the widespread use of microcomputers. VisiCalc was an electronic spreadsheet equivalent to the ledger sheets kept by accountants. One of the major advantages of an electronic spreadsheet was that changes made in the spreadsheet resulted in all other computations being recalculated automatically. This was a great improvement over the paper equivalent. Another spreadsheet program, Lotus 1-2-3, was shipped in January 1983 . This package helped to establish the PC industry within corporations. DOS, the operating system developed for IBM by Microsoft, was begun in November 1980. This operating system became the microcomputer standard operating system and helped Microsoft become a major software vendor. DOS 1.0 was released with the introduction of the IBM PC in August 1981. In many ways VisiCalc, along with the IBM PC counterpart Lotus 1-2-3 (1983), helped legitimate the microcomputer in corporate environments. County X used VisiCalc in a major corporate fiscal impact model (1980) shortly after its introduction into the market.

County X acquired its first PC in March 1982. This occurred eight months after the introduction of the IBM PC. By this time the impact of microcomputers for education was already established by the introduction of the Apple II family of computers. The first Macintosh was acquired by County X in February 1984 (two months after the Macintosh was introduced). Compaq released the first PC clone in 1984. A clone is a computer that runs the same software and operating system as the original, in this case the IBM PC. Prior to 1983, many word processors ran on dedicated equipment where a vendor supplied both the hardware and software. Word Perfect, the word processing software company, released its product in late 1983. Personal computers permitted many different software packages to proliferate and provided many choices to organizations. Administrators eventually had to limit the proliferation of software and hardware choices made by schools and departments by establishing software and hardware standards. County X had not established any such standards by 1982. In the 1982 budget, money was set aside for the evaluation of new hardware and software for possible adoption by County X. This was again illustrated with the early acquisition of the first PostScript laser printer in March 1985. These examples indicate the *proactive* computing nature of County X. This proactive funding of technology, however, did not often occur in the municipal component of the county.

January 1984 marked the introduction of the Macintosh computer. Preceded earlier by the Lisa microcomputer, the Macintosh used an operating system that was graphic oriented and utilized a mouse as a pointing device. The use of a graphic user interface had GUI) significant impact on future microcomputer operating systems such as Windows (Windows 3.0 was shipped in May 1990). Thus the Macintosh led to the widespread acceptance of a graphic user interface.

Milestone Event

During this time period, technological changes were taking place within County X. A major milestone took place in July 1984. The supplier of the corporate computer system decided to discontinue that particular computer system. This decision was unprecedented within that company and had not been anticipated by County X. As noted in Figure 15, County X had just completed a major computer upgrade in the first week of July 1984. With the announcement that this newly installed computer was to be phased out by the manufacturer, County X was faced with a situation where its new computer would now become obsolete. It was estimated, at that time, that approximately twenty programmer years would be needed to convert recently created applications to a new computing platform.

While the salary costs of the programmers were substantial, these salary costs were not the only costs to the district. In addition to the hardware and personnel costs in converting existing applications, new initiatives had to be deferred in order to permit the conversion of existing applications. This resulted in time delays for new initiatives planned by both education and municipal groups. These delays had significant impact on the organization in terms of planned initiatives, increased difficulties with council and board in acquiring funds, and so on. The computer vendor's decision to phase out the computer line, more than any other event in County X, had an impact on later decisions by the school district to move towards a "machine independence" software solution. Machine independence refers to the creation of applications in such a way in order to make movement of these applications to other computer platforms as cost effective as possible.

After evaluating various options with respect to future computing platforms, County X acquired a mini-computer for "research and development" purposes in early 1985. By converting *portions* of existing applications to the new computer architecture, problems were identified and estimates made to the feasibility and costs of converting the remaining applications. This was the first *prototype* application developed by the district. County X decided that this was the best computer platform and existing applications were to be converted to the new computer architecture. Municipal software applications, such as property assessment, were converted first. These conversions were followed by corporate applications such as the financial applications of general ledger, accounts receivable, accounts payable, and purchasing. The student record system was left until the end of the conversion phase. The sequencing of these conversions illustrated the impact of having a treasurer in charge of a computing department that was financially oriented and reported to the Commissioner. The sequencing of the conversion priorities also illustrated that the "municipal versus education" climate continued to exist within County X.

As noted earlier, the decision in July 1994 of the manufacturer of the computer used by County X to phase out a complete computer line which included the computer used by the County had a significant impact on County X. The magnitude, in terms of both time and money, of the impact on an organization of a computer family being terminated resulted in the realization by the district's personnel of the need to develop software that could be more easily migrated from one computing environment to another. This realization was a major change of philosophy and affected subsequent computing decisions, such as the selection of the computing language to be used for future development.

Antecedent Decisions

It was clear from the rapid changes in computing technology that, while the conversion of software would be both expensive and time consuming, it would likely not be the last time conversion would occur within the life of many of the applications. There was a need to reduce the "down the road" costs of conversions that were likely to occur. Some of these decisions are included in the antecedent portion of the study since this is the time that the decisions were *actually* made. They more *logically* fit within the case study since it was in this period that the decisions were actually *implemented.*

Two decisions were made to address the need of creating software that would be more portable and hence reduce the future costs of the applications. (1) C was chosen as the language of development for the SIS project (July 1987). This switch to C was a major departure from the existing practice within County X of using COBOL, which was both the established language for commercial application development and was used for all other previous conversions in County X . Application software *evolves* and therefore it was anticipated that it would need to be converted to some other computing platform in the future. New tools were needed to help minimize programming conversion time and cost associated with these future changes. A computing language which had the potential of reducing the time and effort was required. Lyle and Lloyd made this decision This unilateral decision further illustrated the different approaches used by the municipal and education groups. (2) The SIS rewrite would utilize a data base management package instead of retaining the existing data structures. The move to a data base manager from the traditional file structures solved some data organization problems that had plagued the earlier student package. This constituted a "rewrite" instead of a "conversion" and was in defiance to the Treasurer's guidelines of "no enhancements or improvements" to applications.

One of the problems solved by using a data base management system was that of students attending more than one school at the same time. For example, a student could be attending a regular academic program in one school for part of the day and a vocational program in another school for the remainder of the day. In the existing student record system each student was assigned only one record. This record contained all the data regarding the student. School administrators "worked around" the fact that a student could have only one record (and therefore one school/grade/program) by creating *duplicate records* for such students. This allowed each school to create home rooms, assign marks, and perform the tasks required by that school. While this solution worked at the school level, it created problems at the district level. Because much of the funding for students is driven by student counts, the schools had created "duplicate" students in order to meet their own data needs.

The use of a data base provided a solution to this problem. In order to get around the difficulty of a single student being in more than one school at a time, the student demographic data were broken into a unique "static" record and variable numbers of "dynamic" records (Figure 16). Data that did not change often and were unique (static), such as the student's name, address, date of birth, etc., were incorporated into the static record. Data that were unique to each school, such as room, grade, marks, attendance, etc., were incorporated into the dynamic record. The breaking of data into these two different types of records was a major change in file layouts from the previous application package. This allowed for the single registration of a student for funding purposes, but allowed for the reality of a student attending multiple schools at the same time with each school having the ability to update and change their data on that student.

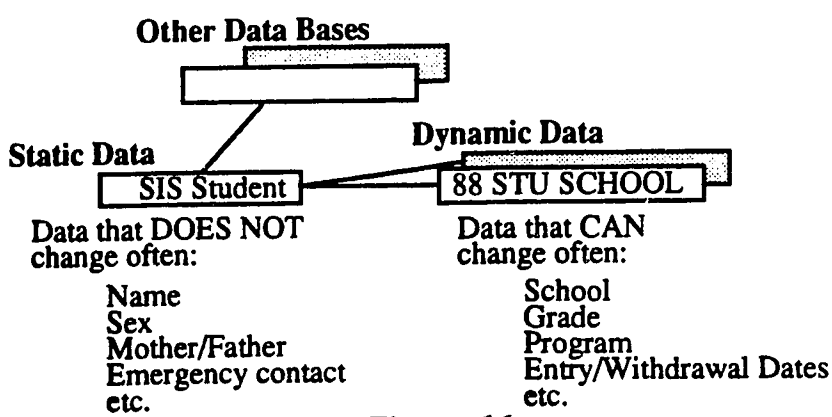

Figure 16
SIS File Layouts

The student attendance system, which became operational in the spring of 1987, was the first student related application to be converted to the data base on the new computer platform. In early 1987 student records were on-line to the high schools, one junior high school, and one elementary school (Figure 27). The student attendance system served as a prototype for the programmers in building a module of the student information system by utilizing both new data base tools and a new computing architecture. This application was written in COBOL rather than C since the programmers felt that to use a new computing language, in addition to a new data base manager and new computer, would be a high risk within the time constraints given. Later, once the new hardware and software were better understood by the programmers, all subsequent modules were developed in C.

Developments Impacting SIS

Alberta Education has a variety of regulations with respect to school facilities and student transportation. Students are eligible for free transportation if they live more than 2.4 kilometers from the school in which they are registered. In the absence of tools to assist in determining student eligibility for free transportation, the schools and the Student Transportation Department had to manually verify each student's eligibility for transportation. This resulted in some students being transported who were ineligible for free transportation while others who were eligible for free transportation were not transported. Operating fully loaded busses and taking the optimum shortest routes posed difficult problems for transportation staff since the bus routes were continually changing and evolving. Clearly, County X required automated tools to reduce the costs of transportation.

A commercial computerized student transportation package was acquired by County X in 1987. This software, which relied on the demographic data of the student information system, was capable of optimizing bus routes and of performing a boundary analysis for the modification of school boundaries. This software was acquired to assist County X in reducing costs of bus runs, as well as improving the boundary planning and school utilization by the district.

Facility utilization posed a related problem. Alberta Education fully funds new school construction only when a school district's facility utilization is equal to or greater than 85%. For any district that is building a new school that does not have a district utilization of at least 85%, the province reduces its portion of capital funding to the utilization level. Local taxpayers are, therefore, required to pay the cost differences between fully funded capital projects and the rate based on district space utilization. Consequently, school districts strove for increased facility utilization if school capital construction was being anticipated. Since County X was expanding with new residential subdivision development and school construction was likely, it was anticipated that specific educational programs would have to be relocated, instructional space would be reduced by facility rentals, and school boundaries modified to improve space utilization in various schools.

In order to be able to analyze both transportation eligibility of students and bus optimization, as well as boundary planning, accurate student addresses were required. In order to achieve this, *two* addresses were incorporated into the new student demographic file structures--a transportation address and a mailing address. The transportation address was declared to be one of the student mandatory fields (a student could not be successfully registered if one of the mandatory fields was absent. These fields included the student's legal name, date of birth, sex, school support codes, school, grade, and program). Since addresses are often abbreviated in many different ways (e.g., Crescent, Cres, Cres., Cresnt, etc.), valid addresses in the entire district were placed into an address data base. When a student's address was entered, the address was validated against the master address file and the correct address name was placed into the student transportation address. This process greatly reduced transportation address errors and allowed the transportation/boundary software to more accurately locate the student's geographic location.

The second address, the mailing address, was required in order to send mail to a student's home. In a county, there are many mailing addresses that make use of post office boxes, rural routes, and so on. This mailing address was required for any mail that might be sent to a student's home. Such mailing addresses did not allow for the geographic location for transportation/boundary purposes. Students with mailing addresses that could not be used for a transportation address were required to receive a "pseudo transportation address." County X had been divided into a system of range roads (running north-south) and township roads (running east-west). Maps and rules for assigning rural addresses based on these roads were distributed to each school. When a student was being registered on the student information system, a school derived "pseudo address" was placed in the required transportation field which specified the geographic location of where the student lived and the mailing address was entered into the second address field.

All schools in County X went "on-line" with access to student demographic information in May 1988. The reason for being on-line by May was to allow time to collect the student demographic data which were required by the student transportation/bussing software as well as for the construction of the next year's student timetables by the schools.

Technology Trends

Specific data requirements for student records changed as a result of more precise information being required by the new software, such as the transportation and boundary analysis software. Parallel to the evolving data needs were the rapid changes in both hardware and software in the computing industry. Hardware and software underwent many revisions and changes in the period from 1982 to 1988. In order to reduce the impact of future hardware and software changes on school districts, educational administrators need to select software and hardware that will not only continue to grow and evolve, but they must also plan for the *rapid change* that takes place. The rapid changes in hardware can be illustrated by the following changes in microcomputers.

The original IBM PC contained an "8 bit" Intel 8088 central processing unit (CPU) running at a clock speed of 4.77 MHz (megahertz--million cycles per second). Since all computers operate in the binary number system (base 2). "8 bits" refer to 2^8 binary digits (bits) that can be directly accessed by the computer. Every computer has a set of instructions that it can execute. These instructions are executed at a rate dictated by its "clock," a timing mechanism that controls the electrical signals in the computer. The IBM AT, released in 1984, was based upon a "16 bit" Intel 80286 CPU and operated at a clock speed of 4.77 MHz. This computer was capable of addressing 2^{16} bits of information at 8 MHz. Not only was the clock speed approximately twice as fast as the original IBM PC, the CPU was capable of addressing much more memory. This allowed for larger, more powerful programs to be run on the AT. The Intel 80386 (2^{32} bits) was released in 1985. The state of technology in late 1991 was an Intel 80486 (also "32 bit", but running at speeds as high as 50 MHz). The Alpha microcomputer purchased by County X in late 1992 was a "64 bit" computer with a clock speed of 150 MHz.

Similar increases in performance occurred in the Macintosh computing family. The Macintosh was based upon the Motorola 68000 CPU (a "32 bit" machine). Like the PC family of computers, the Macintosh family also saw a number of new computers based upon more powerful, faster CPUs. The Macintosh II (utilizing the 68020) was released prior to the study period with the Macintosh fx (68030), and the Quadra (68040) released during or after the study period.

The CPU in the PC microcomputer has a different instruction set than that of the Macintosh. Software written for the IBM PC with an Intel based CPU will not run on a computer with a Motorola 68000 CPU. In the early days of microcomputing, software vendors wrote separate versions of their software for each computing platform, much as County X had to do with its applications software. An IBM PC (8088) with 64K of memory, two 160K floppy diskette drives, and CGA graphics sold for $4,500 US in 1981. A 80386 based PC clone running at 33 MHz with 8 MB memory, a 1.2 MB floppy drive, 200 MB hard drive, 16 bit VGA card and Super VGA monitor sold for $3,500 US in 1991.

The importance of these changes in microcomputers to educational administrators is the *trend* of the CPU towards a constant improvement in both power and speed with a concomitant decrease in cost. This trend of increased performance/decreased price is projected to continue into the future. Educational administrators who recognize this trend can plan for capital replacement reserves, relocation of less powerful, older equipment to schools/departments that have lower computational needs (e.g., "trickle down" of hardware and software from senior high schools to elementary schools), and so on. The fact that more powerful, more cost effective equipment is likely to become available underscores the likelihood that existing applications will move to other platforms in the future. The rapid rate of the introduction of new computer equipment and software is important for education decision makers since these initiatives will compete for other education funds in the budget process.

A second point of significance to administrators is the trend of software vendors to develop applications that run on more than one type of CPU. For example, the spreadsheet program Excel was originally written for the Macintosh. It was later introduced for use with the PC. Microsoft Corporation rewrote Excel using the C programming language to reduce its cost of developing an application for different computing platforms (this is similar to County X's concern for the future of its application software). Administrators, looking to establish standards in software for their districts, could select software provided by a vendor committed to having its software run on a variety of computing platforms. The selection of software capable of running on multiple platforms would enable districts to have more flexibility in selecting alternatives as the computing environments evolved and changed.

Antecedent Summary

At the beginning of the case study there was a background of mistrust between staff members in the municipal and education components of County X. This mistrust began with the original municipal decision (1978) to acquire a computer that eventually proved to be inadequate for the needs of education.

Municipal administrators tend to have a different management style than do educational administrators. Municipal administration is characterized by Mintzberg (1989) as more of a traditional bureaucracy. Education administrators have a management similar to Mintzberg's professional association (Figure 3). This is typified by a consensus approach to decision making. Because of these different approaches, there were fundamental differences in the management styles within the two major administrative units of County X.

In order to provide "economies of scale," County X was reorganized to include a corporate services area. This corporate area included finance, computing, purchasing, transportation, and so on. The Treasurer was placed in charge of this corporate department.

Technology, both hardware and software, was evolving at a rapid rate. New software developments, such as spreadsheets, were being introduced into organizations. Other software, such as data base management software, was permitting new ways of formulating solutions to data problems. This was illustrated by the administrative problem presented by students who registered in more than one school at the same time. The rapid development in data base managers placed pressure on the corporate computing group for different solutions than were typically utilized by traditional computing environments. The County was in a turbulent environment as a result of these changes while the computing group was not only involved with the organizational changes but was being affected by the rapid changes taking place in the computing industry.

Case Participants

This case study covered the period from September 1988 to August 1989. The antecedent events in the organization, the events involving the computing centre, and the events in the computing industry provided a context for this study. The antecedent events provided a background to the events of the study and added depth to the understanding of the events themselves.

There were a number of individuals who took part in this study. In order to facilitate a description of the events, a series of pseudonyms were created. These names are fictitious. The key individuals' names are: **Boris** (the treasurer), **Lyle** (the coordinator), **John** (the school administrator seconded to the project), **Lloyd** (the project manager and author), **Frank** (the superintendent), **Nathan** (a programmer/analyst), **Joe** (the corporate computing manager), and **Rick** (the municipal commissioner and former treasurer).

Lyle: Age 34. BEd (with distinction). Lyle taught for two years before deciding that he wanted to be a full-time computer analyst. After he left teaching, he enrolled in an MSc program in computer science. At the same time he began to work on the student record system (beginning in 1982). Lyle was a highly competent, true "computing scientist" who was proud of his technical abilities. He was equally comfortable working with Macintosh developers, UNIX developers, data base specialists, and "techies" (technical support people). At the same time, he was able to deal with novice computing users and staff at all levels. Lyle was fluent at an advanced level in many computing languages, worked very long hours that include weekends and evenings, but was also a committed family man. Lloyd described Lyle as being a "world class" programmer and one of the few people in the world to have developed a student scheduling package. Lyle was knowledgeable in creating and administering budgets as well as in the management of personnel. He had excellent communications skills and was held in extremely high regard by his co-workers.

John: Age 47. John held a number of positions within County X, as well as with other districts throughout the province. Outgoing and extremely friendly, John was well liked and respected by those that knew him. Originally trained as a school counselor, John had been at a large senior high school for a number of years. He was involved with the existing student record system since near its inception in County X, and was one of the most knowledgeable computer users of this system. When John left the project team to return to his school, the programmers gave him a large rubber stamp with "So far--so good." This stamp illustrated both the programmers' appreciation of his participation, as well as John's normal response to their questions to whether some programming task was complete or not.

Boris: Age approximately 40. Previously a treasurer for other Alberta municipalities, Boris was selected as the candidate mutually acceptable to both the Superintendent and Commissioner. Boris had a reputation for a strong work ethic and for being "results oriented".

Lloyd: Age 41. After completing a B.Sc. degree, Lloyd entered the Faculty of Education due to an interest in computer assisted instruction. Lloyd obtained a Professional Diploma/After Degree. After teaching in Australia for one year, he went on to complete an M.Ed. degree. Following this he worked as a consultant for six years with a large urban district in Alberta. As a consultant, Lloyd worked on the design of a new student information system and served as a liaison between the computer department and the schools. Lloyd originally came to County X in 1979 as an Executive Assistant to the Superintendent to represent the district's computer needs within County X. Lloyd was instrumental in establishing County X's administrative computer program, hired the first computing manager, established the position for Supervisor of instructional computing, and acquired many microcomputers and software packages. Lloyd was the key computing individual within County X.

Frank: Age 51. Frank originally came to County X as the Assistant Superintendent of Human Resources. He became Superintendent of County X upon the retirement of the Superintendent. Known for his strong commitment to schools and extremely friendly personality, Frank's management style was to reach consensus. He worked closely with his management team and with the school principals. Frank was very friendly and had an outgoing personality.

Nathan: Age 31. Nathan was educated in Hong Kong and completed a B.Sc. degree in computing science at the University of Alberta. Nathan was similar to Lyle in that he possessed a genuine interest in computing as a science. Nathan began as a programmer in County X by assisting in the maintenance of the student record system and gradually assumed responsibility for the technical support of the operating system, data base, and telecommunications. He wrote both the student attendance system and scanner interfaces for this attendance system.

Rick: Age 41. A previous employee of the municipal finance department, Rick was invited back to serve as the County Treasurer in 1982. Rick served as Corporate Treasurer until becoming Commissioner for the municipality. An extremely hard worker who constantly worked long hours, Rick's management style was "hands on" as well as being more like a traditional bureaucrat. He valued people's input from any level. Rick was extremely accomplished at completing whatever he wanted done. With respect to computers, Rick believed that computer applications were "built, then left and get on to other things" and hence that software does not need much maintenance or support. Rick also believed that a manager did not have to be "technically competent" to manage a computer operation, but rather that he had to be a good manager.

Joe: Age approximately 45. Joe came to County X from a university computing environment. He replaced a highly competent manager who had excellent people oriented skills and was well liked by the programming staff.

Case Eve[illegible]

The case events (Figure 17) are woven [illegible] threads as the antecedent events--organizational, district computing, and [illegible] industry. The latter category, the computer industry events, is not as [illegible] case study as it was in the antecedent events. While some events did occur [illegible] category, the general trends of increasing speed and capacity of hardware, new and more powerful versions of software, and so on, were already demonstrated.

Case Events

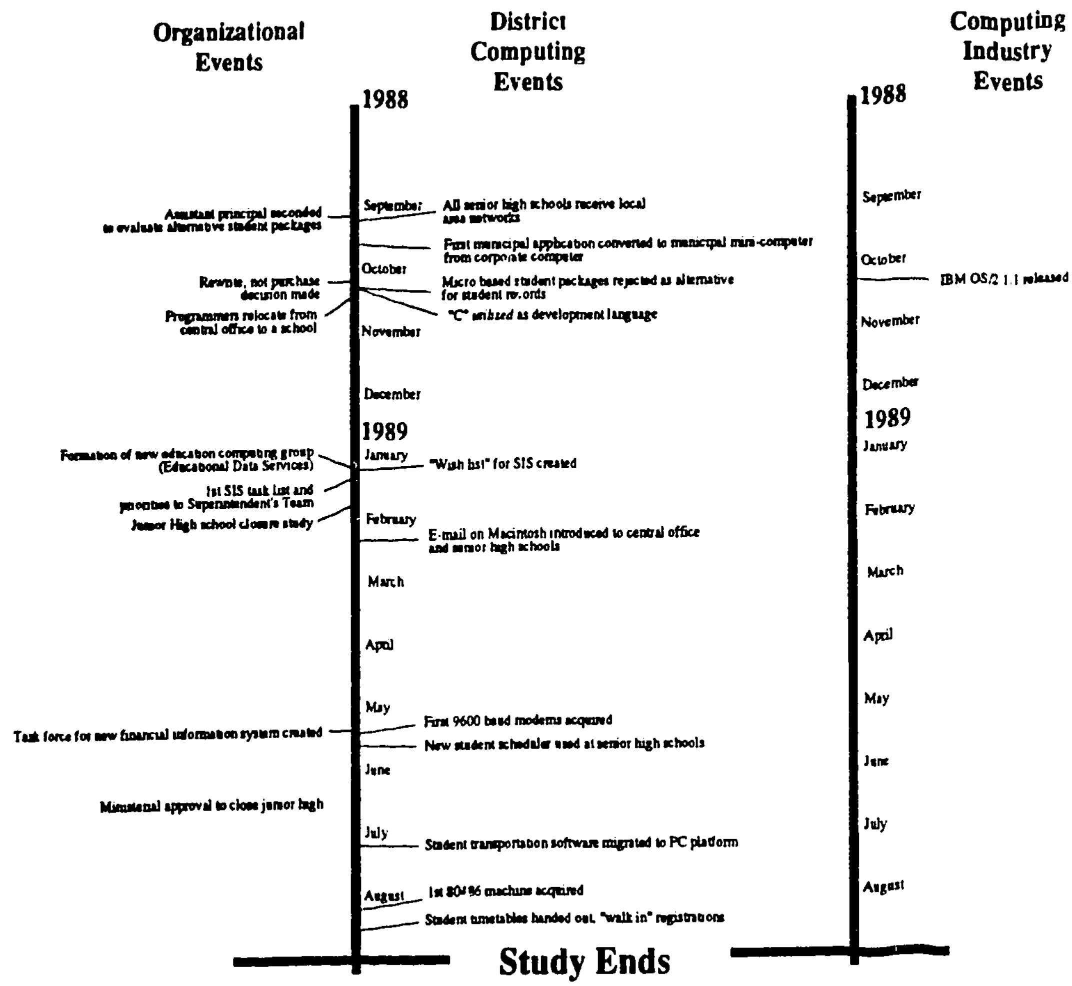

Figure 17
Case Event Timelines

Events in this case study did not occur in isolation. Some events occurred as a result of antecedent events and are better understood in the context of those antecedent events. The computer development group (Figure 12) was a department within an organization. This group was charged with the development of the student information system and utilized development tools, such as information engineering and prototyping, to meet its objectives. Such tools are generally poorly understood by senior administrators, school personnel, trustees, and the parents. In an attempt to improve the success of the project, a person called a *boundary spanner* was used to span the interface with both the senior administrative group and environment.

The organizational *location* of the computing department is the mandate of the senior administrators. Senior administrators control the organizational climate, the resources allocated to the computer department, the funding of the boundary spanner, etc. They must interact with the computer development group as well as with school personnel, central office department personnel, and parents. The *interface* between senior administrators and these groups of people is often well established. A boundary spanner was required as the interface between the computer development group and other groups within the district in order to "translate" the implications of sophisticated techniques of computer development in one direction and the administrative and political implications in the other direction.

The Boundary Spanner

In September 1988 John was *seconded* from a senior high school to the SIS project to represent the school-based users on the SIS project team. Since there were extremely tight timelines for the phasing out of County X's computer, it was felt that John could assist the programmers by working directly with them. Since the timelines were very short, senior administrators approved a one semester secondment of John, a key school-based user of the student record system, to work on the student record conversion project. John's first priority was to evaluate existing microcomputer based packages to determine if there was a suitable package to replace the existing system. In May 1988, the County's senior high principals stated that any new system, whether microcomputer based or mainframe based, had to be *at least as functional* (from the users' point of view) as the system already in use. These principals did not want a student records software package simply because it was microcomputer based or because it was cheap.

John's secondment came as a result of Lloyd's desire to ensure user representation on this project, especially since the project was controlled by the Treasurer. Programmers operate in a highly technical environment where decisions are made on data structures, efficiency of code, and so on. School administrators operate in a student and teacher oriented environment where "service to students" and instruction reflect the environment. Since these are two different environments, Lloyd felt that an administrator who "lived with" the programmers would be able to establish relationships with the programmers and begin to understand the issues and problems in that environment. Lloyd believed the presence of a user on the project team would improve communication and permit decisions to be made in shorter lengths of time than if that person was not present. The shorter decision times would facilitate in meeting the project's short timelines. Another reason for establishing the boundary spanner position was to ensure that interests of the end-users would be maintained in the project by providing an independent source of project information.

Frank felt that the programmers would also benefit by having someone represent "how it was done in the school". When recalling John's role, Frank noted, "He brought 'school friendly' to the project. John had good communication skills and brought to the whole project how software needed to be from the schools' point of view" (Personal Interview, April 1992).

Lloyd had "built the case" for the boundary spanner position in the budget planning process in February 1988. Based upon his experience in previous computer projects, Lloyd knew that there was a tremendous amount of work involved in conversions of software. If the work load associated with software conversion was "added on top of" a school administrator's work load, both the conversion project and the administrator's effectiveness in the school were likely to suffer. Funds for such a secondment ensured the focused attention on a computer conversion project with tight timelines and did not penalize the administration of the school by adding additional tasks to individuals within the school.

John was the most knowledgeable computer student records administrator within his school. In his school, John's responsibilities included timetable construction, report card production, and day to day duties such as student attendance and student discipline. He did not begin working "full time" with the programmers when schools opened in September 1988. Like many high schools, John's school had a large number of students register at the beginning of the fall term. John was needed in the school to supervise the registration process. As a result, John began work with the computing group near the middle of September. Anticipating John's arrival on the project, the SIS project team had been working with potential microcomputer software vendors and had either already acquired evaluation copies of software packages or had established contacts with potential vendors.

John began the evaluation of microcomputer software in the office space shared by both the corporate programmers and SIS programmers. By October 1988, John decided that none of the packages "would do what we were already doing" (Personal Interview, February 1992). He summarized some of the specific findings in a report (Technical Evaluation of the Micro Scheduler, Appendix B). John was empowered by the Superintendent's Team and the other senior high schools to make decisions with respect to the various software packages. John "polled" various key schools with the progress of the evaluations and the various strengths and weaknesses of the software packages were discussed amongst the school administrators. The senior high school principals reached consensus that none of the microcomputer based packages were "as functional" as the existing system. John brought this consensus to the superintendent's team with the recommendation not to proceed to a microcomputer based system (October 1988). This resulted in the decision to *rewrite* the application. Without John actually performing the evaluations and communicating progress and results to his colleagues, it is unlikely that this recommendation could have been arrived at in such a short time frame. The classical approach to computer development was to have "users signing off" their concerns (like Carlsen and Lewis, 1979). This would not have worked with the short timelines of this project.

The rewrite project, left by Joe until the end of the conversion task list, became a high priority with very severe time constraints. John's mandate was modified to work along with the programmers on a rewrite of the software and to represent the needs and interests of the schools. To this end John's secondment was extended from the original one semester to two semesters. In order to facilitate these tasks, John moved out of the space he shared in Lloyd's office and moved in with the programming staff.

SIS Team Relocation

By early October, the SIS programmers had expressed *serious* concerns about the magnitude of the tasks to be completed and the time available to complete these tasks. As noted earlier (Figure 14), seventeen assumptions were outlined describing what was necessary for the successful completion of the project. Assumption 7, that staff would not be relocated to other "emergency" tasks and removed from the project, was not being met. Joe reassigned members of the SIS project team almost daily to "emergency tasks" in other areas such as payroll, general ledger, and so on. John stated, "It was a hassle getting anything done. Every time we started to work on the course master or timetable, they (the programmers) got pulled off and put on something else or had to attend some meeting" (Personal Interview, March 1992). If the rewrite timelines were to be met, steps were required that would permit the programming staff to remain focused on the rewrite tasks.

Boris undertook an examination of how the reallocations of the programmers occurred. He found that most emergency tasks were a result of individuals in user departments approaching the programmers directly with software concerns that needed to be rectified. This was not unusual since good working relationships between the programmers and users had developed over a period of years and often the programmers were the only people in the organization possessing a working knowledge of how various computer systems functioned. Boris issued a directive, confirmed by the computer steering committee, that all further requests from user departments would "have to come through channels" and that this return to a more rigorous procedure would be supported by senior administrators.

John felt that many of the interruptions were a result of programming staff being taken away from the education application (the rewrite) and back to municipal computing problems. He believed that this situation was "just a continuation of a municipal treasurer and municipal oriented manager continuing not to take education seriously" (Personal Interview, March 1992). The "municipal versus education" and "we-they" perceptions were becoming amplified by the time constraints of the project timelines.

In order to minimize interruptions, Lloyd and John sought a relocation of office space for the programming team. The rationale for wanting to move the programming team out of the central office administrators building was to reduce the informal "drop in" traffic by various user groups and to foster the focused attention to the project tasks. Vacant space was located in one of the elementary schools. Desks and chairs were borrowed from various schools and departments and were moved to the school. The "move" took place in October 1988. The SIS programmers continued to maintain their desks, phones, and workstations in the central office, but they did not report there for work at the start of the day. The programmers arrived for work at temporary location in the school. To minimize the interruptions by phone, only one phone line was installed at the new location and the telephone number was not generally distributed. User departments or schools that needed to contact one of the programmers were required to contact either the project manager or supervisor of operations. If one of these two individuals agreed that the problem was an emergency, the call was logged and a message was passed on to the programmer. The programmer then returned the call to the user. This new procedure applied to all user departments including senior administrators on the superintendent's team.

Lloyd and Lyle prepared biweekly status reports and provided the reports directly to the Superintendent's Team. This built accountability for the project team with the superintendent's team without having to go through Joe.

The biweekly status reports (Figure 18) became the reporting mechanism between the project team and senior administrators. The following report indicates the *type* of information communicated in the early stages of the reporting. To maintain confidentiality, school names, people's names, and positions appear in *italics;* they have been modified to ensure anonymity.

SIS Reporting
for the Period to 22-Sep-88
(Addendum to 29-Sep-88)

1.0 Highlights

•reporting will be for every 2nd and 4th Monday. Reports will include tasks to midnight of the Wednesday previous. These are reviewed on Thursday in time for inclusion in the Superintendent's agenda.

•*school* relocation proceeding:

Target date for facility alterations Oct. 3. AGT must trench new lines - target date Oct. 6. The trenching date has now been scheduled for October 28. Options such as a multiplexor instead of 513's, lease costs of such versus original plan, contracting trenching, etc. are being pursued.
Terminals have been ordered. 4 desks loaned from *school #1*, 2 from *school #2*.
•work proceeding on rewrite of student scheduler. *Lloyd and John* are meeting with *microcomputer vendor* personnel Oct. 12.

•projected overrun 56 days. This is primarily due to two factors: *Consultant* only available for the average of 1 day/week to January 89; time spent on getting *micro package* up and running. This is not yet a cause of undue concern.

•discussion took place as to who should have access to the marks in the Phase II system. Should Central Office be able to access current marks and not just historic marks? Should one school be able to "look" at the marks of their students attending another school half days and vice versa (i.e., in a multiple registration, one school could look at the other's current marks). Is it desirable to have one report card for these students and not two. Do central users such as Revenues need to look at current course registrations for fee collection purposes? It was decided to build flags into the system to allow any and/or all of the above (different classes of users with different access).

•speed of lines has been a problem at some schools (*Sch*$_1$, *Sch*$_2$, & *Sch*$_3$ are running at 1200 baud. *Programmer*$_1$ is monitoring the situation and will escalate to AGT. If this does not work, then either clean lines and/or error correcting modems may have to be acquired.

•task 100 (support to existing) was budgeted 40 days in Part A. 20 have already been used in one month. Part of this was expected as September is a start up month. Other activities such as using SIS live for revenue reports (Age/Grade/Sex, non-resident lists, school district lists, etc.), enrollment by school, etc. have added work to refine, fix, or create reports and are charged to task 100.

2.0 Decisions

2.1 Funding. Decision required as to where funds are to be charged and which are to be charged. Position of *Computing Manager* is as per his letter (attached), *Lloyd* as per my letter (attached).

2.1.1 Points 1 & 3 of *Joe's* letter: Currently the costs are temporarily in department budget subject to year end. Prediction is that sufficient funds to cover. If not, will be brought forward.

2.1.2 Training. In the same letter, *Joe* wishes someone to be acquired for training.

2.2 Other. It is proposed that much of the detail of this report not come forward after the next few status reports are presented (Internal correspondence, October 1988).

Figure 18
SIS Status Report

This status report, along with other reports and correspondence (Appendix B) illustrate the general organizational climate during the last quarter of 1988. The "temporary relocation" to a school for the SIS team became "permanent" in January 1989.

Educational Data Services Formed

In September 1988 the computing group was a "corporate" department charged with the responsibility of providing computer services to both the education and municipal components of County X. In January 1989, this group was split into a municipal group under Joe's direction and an education group under Lyle's leadership. It was noted earlier that Joe's development style was similar to the Carlsen and Lewis style of "user signoff" prior to development. It was further noted that there were organizational "stresses" between the municipal and education components.

One of these stresses was the combination of leaving the student information system conversion, a uniquely educational application, until the end of the conversion list, coupled with Boris' philosophy of "no enhancements--rewrite only". As noted in correspondence (Appendix B), key staff were being removed from the SIS project to other administrative tasks. This was summarized in letters to the Superintendent's Team between October 1988 and January 1989.

> •*Lyle* has spent 15 days since September on non-SIS material (7 were projected). It is anticipated that the amount of time spent on non-SIS will diminish as the new technical support contractor picks up tasks. *Programmer*$_2$ has spent 8.5 days since September on non-SIS material. It is expected that this will halt (Superintendent's Team letter, October 19 1988).
>
> In the past this was offset by programmers working overtime and the contract programmer working at the front end of the project. It is not realistic to expect that programmers can continue to work the same amount of overtime since they are becoming "run down". The rate of being "siphoned off" to other projects and tasks instead of SIS has continued (Superintendent's Team Letter, November 30 1988).
>
> •since Dec. 15, 6.0 days for *Programmer*$_1$ to Payroll, ~22 days for *Lyle* for department split and budget (Superintendent's Team Letter, January 25, 1989).

Lyle, the key programmer in the rewrite, spent 15 working days from September 1988 to October 15 on tasks not related to the SIS project. Since there are usually 20 working days in a month, Lyle was spending approximately 50% of his time outside of the SIS project. This rate was similar in the November 30 update. The rate had *increased* to nearly full time at the January 25, 1989 update. Joe's "official" position in January 1989 indicated that the SIS project was relatively on target. Lloyd and Lyle indicated that the "'Doability Probability' of the project for Mar. 1 was "very low" (Superintendent's Team Letter, January 25, 1989). Lloyd coined the term "Doability" to refer to the probability of success of the project. However, what happened "behind the scenes" was quite different.

There was a high level of frustration expressed by the Superintendent's Team that a high priority education project, one that had already seen the development team moved to a school to minimize interruptions and which was crucial to the operation of the schools, was being put in jeopardy by Joe's reallocation of key individuals to other projects. The SIS programming staff felt this frustration also. As a result of continuing to work great amounts of overtime to meet the project deadlines, most of the programmers were actively seeking employment elsewhere.

Lyle was a key person in the SIS project. He was the leading internal expert in C, was an "educator turned programmer," was experienced with the existing system, and was the person charged with the task of rewriting the student scheduler. Frustrated with the working environment, while still attempting to keep the project on time, Lyle applied for and received job offers from outside the organization. The potential loss of Lyle placed the entire project in a serious, if not fatal, condition. His search for alternate employment reflected Lyle's high frustration levels.

Lyle's frustration levels with both Boris' management style and with Joe did not occur "overnight." Lyle began to experience frustration shortly after Boris arrived. Boris had instituted a procedure that all requests for computer services were to be submitted on a specific form. This form would then need to be approved by either Joe or Boris. This procedure was supported by Joe, but was viewed by the programmers as "busy work". Shortly after this procedure was mandated, Boris requested that some computers be set up but did not make use of his own approval form or procedure. Lyle's response to Boris gives some insight into the tensions present as early as 1987:

> "...you made a verbal request to have your IBM AT...and Macintosh to be connected...Since the scope of the request was not immediately apparent you were asked to fill in the standard request forms in order to have your request fleshed out more fully.
>
> I was surprised and disappointed when you walked out on our discussion angered about having to do this paper work. You pointed out that you have done 'considerable fighting' on our behalf to provide our department with the resources it needs and that if we were 'good marketers' we should realize the importance of helping you and not impeding your requests for service.
>
> I feel your anger was unjustified. I thought you, more than others, would realize the importance of following the procedures you've had us put into place. I am concerned that you, as General Manager, would expect me to short circuit the same procedures you expect us to hold other departments to. You have placed me in a very uncomfortable position in expecting me to bypass those procedures and policies agreed to by the Steering Committee" (Internal letter, November 1987).

Staff continued to leave Boris' departments for employment elsewhere.

Another significant event took place in November 1988 with the planned reorganization of the computing department. Joe proposed that the computing department hire a manager of operations to fill the vacancy created by a programmer/analyst leaving (November 1988). Lyle had been resisting this particular structure within the department since it would have created five management positions in a department of thirteen individuals, a level of "overhead" that he felt was unjustified. Lyle's view that, "We need more workers, not more bosses" (Personal Interview, November 1988) summarized Lyle's feelings. When Joe decided that this organization structure would proceed and the new position was advertised, Lyle went directly to both Frank and Rick with the concerns that the project would be compromised with the new administrative structure. These visits led to a series of events that resulted in the formation of two separate departments.

Upon learning of the visit to Frank and Rick, Boris called both Lyle and Joe into a meeting. At this meeting Boris demanded a formal apology from Lyle. After the meeting, Lyle summarized his feelings in this way:

> In our conversation yesterday you asked me for a formal apology to you, the commissioner, and superintendent. Since I don't feel I've done anything wrong that requires an apology I cannot do that.
>
> I went to the superintendent and commissioner out of concern for the management complement in the department. Even after our discussions yesterday, I am still not convinced that we need 5 senior management positions plus 1 supervisor for a staff of 13. I still believe that this is too high a ratio....If you want to fill the new position of Operations Coordinator with a person whose skills are predominantly management oriented then I concede you have that right. But you should also concede that I have the right to express my concerns when I believe you are making an error in judgment. I know you don't approve of my actions and, as you said, you ***'can take me out'*** but I hope you would not dismiss me for doing something that I believe (and still believe) needed to be done...
>
> Also, I feel it was unfair to try to discredit my management abilities by holding me responsible for the 60 day overrun of SIS as well as being behind in the installation of the operating system software....I do not want to jeopardize the integrity of the County's data plus the availability of our system by having someone not familiar with our system do it...
>
> I think it was also unfair to accuse me of being responsible, to a high degree, for the current problems with the department. If there is a lack of support for the department from the senior administration and other departments then I think that's a bigger problem than just one person can create. The department has been through a difficult time in the past few years due to the high staff turnover. I feel that I have done my best to hold the systems and the people together-but, from our conversation, I know you don't agree. May I tactfully suggest that before laying much of the blame on my shoulders you weigh yourself in the equation as well. If my 'negativism' has caused the high turnover in the department (8 out of 13 in the past 2 years) then how do you explain the similarly high turnover in finance? I'm sorry I can not accept the guilt you are trying to burden me with-it has to be more than just me..." (Internal Memo, November 1988)

Lyle's feelings were not isolated. Lloyd shared his feelings with his supervisor as follows:

> Issues are: accountability
> leadership

Boris has had total control over Finance & Information Services...Boris chose to go for the original IS mandate in '84 in spite of objections. This was reaffirmed on his part by the selective tailoring of the consultant's study. The original RFQ was comprehensive, but the highly edited version assigned to the consultant reflected only those issues that Boris chose to focus on for his own reasons (not based on any mandate from the steering committee).

In December 87, when it was noted that there were concerns with the charge backs to Education, nothing was done by the Treasurer to ensure proper accounting. This is the same treasurer that pushed for "program budgets" and created tremendous amounts of work for departments to ensure highly detailed accounting. It seems that this philosophy applies only to other departments. This occurred in spite of the fact that the Board in the budget process wanted better accounting. The consultant's "mandate" was again the initiative of the Treasurer.

The "reorganization" occurred and was not presented to the Supt.'s Team, even as information. This would result in 5 management positions costing over $250,000/annum - with no job descriptions, no statement as to how it would improve the present situations, no statement as to how it would assist in meeting the deadlines, and so on. The "team" only seems to apply when Corporate Services wants departments to do things "their way".

Splitting will not change this management orientation by the General Manager....Performance is required in programmer contracts controlled by Information Services. It seems, however, that when performance is the issue with management staff, the parties wishing such assurances are "negative" or "not team players".

Corporate Departments received computer lines for secretaries and managers, IS sponsored in services on mail, and so on. The Superintendent and Associates received no such preferential treatment.

The FIS rewrite cost an enormous amount of money. Most departments keep their own books and express opinions that the reports they receive are late and have accuracy problems. This project was entirely under the control of the General Manager.

High staff turn over continues to be a problem. In the assumptions for successful completion of the SIS rewrite this was one of the key issues. Not only is it not apparent how programmers were protected against burnout, they continued to be reallocated to other tasks. The reason for the SIS team relocation was to minimize these interruptions. While successful in reducing outside interruptions, those from within the department continued to occur.

Payroll was 80% converted to the VAX prior to the programmer leaving. Lack of "bench strength" for the project was apparent at that time. Enhancements were then made to the existing system without also being implemented on the converted system, jeopardizing the entire work done on Payroll. Most municipal applications were converted first.

There is a higher than average turnover. Many of the personnel in conversations expressed the management style/situation as one of the main reasons for leaving (Internal Memo, November 18, 1988).

The deep feelings of both Lyle and Lloyd were widespread within the programming group. One of the SIS programmers said, "If Lyle had left, it was just a matter of time before the rest of us left" (Personal Interview, March 1992). Lyle's departure would have seriously affected the SIS project. The exodus of many of the SIS programmers would have halted the SIS project completely. Discussions between Frank and Rick led to a decision to form two separate computing groups--municipal and education (January 1989). The municipal group would continue to be structured as it was while the education group would be managed by Lyle.

John had vivid recollections of the times around Christmas 1988 and early January 1989. One situation was also recalled by a programmer (referred to as Doug). It involved a discussion on one particular Monday morning in the school where the SIS team was located:

> John: I remember that day everybody sat and sulked in Room 3.
> Doug: I remember that too!
> John: It was a depressing day. I was in a panic as to what we would do for the scheduler. If Lyle left, then everything was down the tubes. Could we buy a micro scheduler? We couldn't build one without Lyle. We were worried about where to head.
> Doug: All of us just sat around and wondered where things would go. It was really depressing.
> John: Lyle asked me what he should do--if he should take that job offer or what. I told him to do what he thought he had to do.
> Doug: That's what I told him as well (Personal Interview, April 1992).

The emotional intensity experience by John and Lyle was still evident three years after the event took place!

> John: I remember when Frank told us that they were going to split the computer department. Then Lyle told us that he was going to turn the job offer down and stay to finish the project! We sure had to play 'hurry up' on the scheduler...We were really lucky--we had a good team. What an amazing group we had then! (Personal Interview, April 1992)

Meanwhile, Boris brought forward a document (January 1989) dealing with the reallocation of the assets and reporting structures. Included in this document were modifications to the mandates of the Information Services and Education Data Services in such a way to provide both municipal and education direct control over computing resources required for each organization. Rick had expressed a concern that he did not have enough control over what the computing departments were doing. The rationale was to allow each unit to have independent control over the plans and priorities for computing. As a result of this document, the education group was to provide support for SIS, student transportation, facilities, Human Resources, and education office automation. The municipal group was to provide support for finance, assessment, taxation, utilities, accounts receivable, recreation and parks, Council, Board, and municipal office automation.

The "vision" of the municipal mandate was reestablished at that time. The corporate/municipal group was to "do as much local processing as feasible; contract out computing requirements where feasible; and operate a central facility with the capacity to ensure central data integrity and networking capacity" (Confidential Internal Report, January 1989). Staff were divided between the two groups with the SIS project team and the computer operators becoming the education computing group. Schedules were drawn up to reallocate the existing infrastructure with education taking over the mainframe ownership. This asset was purchased by internal financing with education "buying out" the municipal portion of the mainframe computer.

Frank's recollections of the events surrounding the formation of two separate computing groups demonstrate a different aspect of the events. He noted:

> There was a high level of concern on both sides as to who was getting the most service from the data services group. The municipal people thought that education was getting too much while the education people thought that municipal was getting too much.
>
> There is a difference in management styles as well. I'm more adaptable and can make various situations work. On the other side he has to 'have control'--he didn't feel that he had the final say on what was happening. He had other problems as well. Education had a fair amount of central control while in the municipal there were departments running on their own in many directions. He finally made up his mind that he wanted his own people to meet their own needs (Personal Interview, April 1992).

County X had previously established a steering committee charged with establishing priorities for the organization. Frank noted that this committee did not meet its goals: "The steering committee that was set up to establish priorities didn't really work. How do you compare a student system for us and an application for engineering? They are completely different but are both important for each group" (Personal Interview, April 1992). The structure established for dispute resolution was not viewed as being effective.

The perception by senior administrators on both sides of the organization that they were "not getting their fair share" was widely shared by middle management. Few department heads or principals were given input into documents, such as the conversion timelines. These timelines outlined when various major software conversion were to take place. Coupled with this lack of input was the constant reallocation of resources within the computing group to meet emergencies. The lack of control by senior education staff over project pacing eventually led to a decision to have payroll and human resource functions removed from the corporate computing group to be replaced by a "stand alone" microcomputer-based package (September 1989).

Referring to the Hersey and Blanchard leadership style model (Figure 6), it is evident that there was a mismatch between the "readiness" of Lyle and his group and the "management style" which was brought to bear on the situation. Lyle and his group were high task and high relationship (S2). Boris and Joe were high task and low relationship. The low "follower readiness" can be demonstrated in this model.

As noted in the literature review, Mintzberg described the different management styles--the municipal traditional bureaucratic and the educational professional organization. These different management styles resulted in a developmental group that was in a turbulent environment. Frank concluded, "I believe that the split was a natural evolution of an organization. As it gets bigger and more complex it is just too much for a single group to support" (Personal Interview, April 1992).

Programming Language

A key issue of significance to administrators involved in supervision and long term planning of computing environments is the choice of the most appropriate computing language. Traditional administrative computing has been done using a computing language called COBOL. Applications written in this language have tended to be highly specific to a given computer product line. The conversion of these applications to other computer lines or manufacturers has often proved to be a major undertaking. A language other than COBOL was required to provide the flexibility for applications to be migrated to other computer equipment.

In mid-1987, Lloyd and Lyle decided that all *new* applications should be written in C. Prior modules had been written in COBOL, largely due to directives from Rick and Joe. Starting in October 1988, all modules of the SIS project were written in C. John's arrival as part of the team and the physical relocation of the programming group prompted Lloyd and Lyle to decide to ignore the COBOL directive and use C as the development language. They felt that the change in programming language would better meet the long term goal of machine independence. They also believed that neither Boris or Joe would support the strategic move to change development languages. As a result, there was a high level of stress and dissatisfaction between Joe and the SIS team.

C was a language designed to improve the portability of software between different computing environments. Programs written in C have the potential to be run on more than one type of computer without a great deal of change. This language had the potential to minimize the impact of future changes in technology. Thus C was the computing language utilized for the SIS rewrite project. County X was the first school jurisdiction in Alberta that made use of C as the development language. In anticipation of the need to be able to convert the application to different mainframe computers or microcomputers, the success of using C might be of great interest to people developing computing applications in education.

Machine Independence

As noted previously, the decision by the computer vendor to "phase out" a family of computers was a milestone event that resulted in a decision to move towards becoming less dependent on any one hardware or software vendor. The move to a computing language, such as C, that made future software conversions ("porting") of an application to another computing platform was one of the steps in this process.

The major components of any computing application that are *vendor specific* include the operating system, the data base package, and the screen handling package. The operating system is the software that controls what the computer does and in what order. Many data base packages are proprietary to a given machine architecture. It is quite common for these data base packages to only run on one type of computer. Fortunately, there are data base packages that will run on many types of computers. The screen handling package is the software that controls how and where items appear on the screen; some are proprietary while others are not.

To enable County X to be less dependent on a particular operating system from a specific vendor, Lyle and Lloyd selected UNIX as the future operating system. UNIX was designed to run on a number of different computing architectures. This "statement of direction" committed County X to what is commonly referred to as an "open system" approach to computing. An open system refers to an initiative by a consortium of computer manufacturers and universities that formed the "Open Software Foundation"--OSF. By committing to an "open systems" approach to computing, Lyle and Lloyd sought to establish a computing environment wherein applications could be developed which could be run on a variety of different computers.

Moving to an "open system" had some administrative advantages over staying with a proprietary environment. Open systems, for example, minimize the risk of being stranded due to the "phasing out" of a particular computer family. If an application ran on a computer that was "phased out", the district could select another computer without consideration of a major investment in rewriting the software. In this instance it meant that County X was less vulnerable to future changes in the turbulent and rapidly changing world of computing. In the future the district would likely have lower costs by eliminating costly conversions of application software to other computing platforms. A second advantage was the predicted cost savings as a result of competition among vendors. Since many manufacturers would be selling computing equipment with the same operating system, it would be unlikely that any one supplier would be significantly more expensive than other suppliers. In proprietary environments customers would be committed to paying whatever license increases are set by the vendor.

The data base management software was the second proprietary area to be considered for machine independence. Large educational clients, such as Alberta Education, have their student records system developed for a data base application that only runs on specific large mainframe computers. A data base that could run on a variety of computers was required to be independent of a given computer supplier. Due to the short timelines for the SIS project, Lyle decided in January 1989 to continue to use the existing data base manager in order to meet the project completion dates. At the same time, he decided to begin acquiring data base packages that ran on a variety of different computers. This acquisition phase began in late August 1989 and resulted in a number of different data base managers being acquired and examined. An interim data base package that ran on the UNIX platform was selected in October 1991.

The third component of the SIS system that was proprietary was the "screen handling" software. This software was, again, specific to a particular vendor's operating system and controlled the data that appeared on the screen, window shapes, colors, cursor movement, and so on. A package that could handle all required terminal interaction, but would run on a number of different computers, was required. Due to very short timelines, County X again decided not to proceed with an "open systems" windowing package for Phase II of the project. Like the data base manager, the existing windowing package was chosen for Phase II, with a commitment to examine other windowing packages in the future.

Since the commitment was to move to an "open" operating system, to acquire a new data base manager, and to acquire a new windowing environment, all SIS programs in Phase II were written to minimize the dependency on specific features in each of the three areas. Lyle established a series of computer library routines that could be used by all programs in lieu of the programs using vendor supplied system routines. For example, all existing programs made use of a sort program supplied with the operating system to sort student data. Since the specific sort program from that operating system would not be available in the future, Phase II programs had to be created using Lyle's library sort program.

There were many such programs that would normally be provided by the software vendors that could no longer be used when machine independence became the development philosophy. Because many of the student reports were written in the reporting language of the data base manager, these report programs all had to be rewritten in C and use made of the utilities of the library programs developed in Phase II. The strategy of machine independence had very far reaching impacts into how the programmers wrote programs in order to promote the future "portability" of the software.

Lloyd wanted "machine independence" to apply to more than just the host mainframe computer. Machine independence on the microcomputer side of the project was more easily established. Productivity software, such as word processing, spreadsheets, and desk-top publishing comprised the majority of the software used on either the Macintosh or PC computers in the district. Because of this situation, Lloyd selected software that ran on both platforms. Such software would ensure that the district's word processing files, financial models, and so on would be safeguarded if particular computer manufacturers changed their computer architectures in the future.

In summary, Lloyd and Lyle decided to reduce the risk of future technology changes by adopting a "machine independent" philosophy. This resulted in the programmers structuring their programs in such a fashion to reduce the reliance on any operating system, data base manager, or screen handling software. This philosophy committed County X to acquiring and examining alternative packages that would operate in an "open" environment. Lloyd also decided to acquire microcomputer software that ran on more than one type of microcomputer in order to protect County X's long term investments in data and files developed with microcomputer productivity packages.

The Scheduler

One of the major components of a student information system is the senior high school course "scheduler"--the program that creates a student's timetable from a master course timetable and a student's course requests. County X's student scheduler, prior to the SIS project, was a proprietary package developed in the 1960s in California. This scheduler later became part of the student records package purchased by County X. This same scheduler became incorporated into another mainframe software package acquired by another large Alberta district and also became the scheduler used in a widely installed PC based student package.

The original student package was written in "Assembly language," a very "low level" computing language. In the era that the scheduler was developed, Assembly language was one of the few languages that gave the control and machine speed necessary to perform computing intensive tasks such as scheduling. County X invested a number of weeks in early 1980 in an attempt to "reverse engineer" the scheduler to establish how the program constructed the student timetables. This attempt was unsuccessful. The scheduler remained a "black box", not only to County X but to the many vendors and school districts contacted on the scheduling problem. Nothing existed in the available literature on student scheduling and no one could be found to help explain how a student scheduler worked. The move to machine independence required that this crucial set of programs be understood and rewritten.

There were a number of operational problems with the existing scheduler. One of the major deficiencies was the scheduler's slow speed. For the district's largest school (approximately 1,800 students) the scheduler took approximately 1 hour of CPU time (approximately 8 hours of elapsed time, depending upon what other tasks were running concurrently on the computer). Because it required a large amount of central memory, the scheduler could not be run at the same time as other interactive terminal sessions. This resulted in the schools being unable to run the scheduler during normal office hours. In a typical "scheduling run", an administrator would submit the job one day and examine the output the following day. Since it took many "runs" to build a school schedule, this process took up to two or three weeks to complete a basic timetable.

A second deficiency was that the existing scheduler created student work loads that were not equally balanced between semesters. For example, a student might receive all their "lighter" *elective* courses in the first semester with all the "heavier" *required* courses in the second semester. This resulted in a high level of negative feedback from students and parents who felt that the timetables could be better balanced over the entire year.

A third deficiency was the lack of integration between the various student software modules. Scheduling in a school occurred in two major ways and at two different times--the master timetable and the scheduling of the individual timetables was undertaken in May of each year, followed by the "walk in" registrations that took place during school opening in late August and early September. Since each module (i.e., scheduling, registration, marks, attendance, etc.) was designed to "stand alone," an administrator or counselor was required to register a student, exit the registration module, enter the course request module to enter the student's course requests, exit to enter the scheduling module, and so on. With the entrance to or exit from each module came the "overhead" waiting while the computer loaded the program into memory. The net result of this approach was many time delays when administrators registered and scheduled new individual registrations.

A fourth deficiency had to do with the many "work arounds" that needed to be performed with various packages. A "work around" was a procedure that had to be put in place to circumvent deficiencies in the software capabilities or to permit exceptions to the normal processing events. Two such "work arounds" that caused extra work for the schools were cases involving students who worked part-time and were not available to take classes at certain times of the day and report cards that contained a mixture of semestered courses and full year courses. These deficiencies will be discussed in more detail later.

Lyle's design goals for the scheduler were: to became non-proprietary, to have programs that were well understood, to have documented algorithms used to derive the timetable, to have balancing of student course loads between semesters, and to have faster execution times for the major timetabling tasks.

To address the issue of balanced semester loads, Lyle and John devised a scheme of *course weights*. Senior high school courses require a certain number of hours of instruction for each course. For example, a 5 credit course requires 125 hours of instruction per year. Lyle modified the data definition in the master course file to store a "weighting factor"--an integer that could be used by the scheduling algorithms to assign approximately equal course loads to each semester (Figure 19). Lyle allowed for the weighting to be an integer between 0 - 256. By convention the school administrators adopted John's proposal for weighting--a 5 credit required course would have a weighting factor of 8, a 3 credit required course a weight of 6, a 5 credit elective course a weight of 4, and a 3 credit elective course a weight of 2. English 13 (Figure 19) is a 5 credit course and is assigned by the school to have a weighting factor of 8.

```
CRSMST-1                 91-92          Test School
                  Course Master Maintenance

                 Course Number: 1115
                        Status: A
                    Long Title: English 13
                   Short Title: ENG 13
                        Credit: 5  Weight: 8



                    Department:
                   Course Type: R
                         Level: 4 Senior High
                         Exam?: N
            Alberta Ed. course?: Y
```

Figure 19
Assignment of Course Weights

By designating the weight in the "course master" data base, any student request for a course obtained the weight associated with that course. The scheduler software uses these weights to balance a student timetable between semesters. To illustrate this balance, a fictitious student, Edith Fantasy (Figure 20), is scheduled to take two required courses (Social Studies 20 and Math 20) in the first semester along with two electives (Accounting 20 and Food Studies 20) for a total of 20 credits. The second semester has Edith taking five required courses for 19 credits. In this case there was a relative balance of loads between semesters.

CRSREQ-1 — 91-92 Test School
Student Course Request Maintenance

School:
ID: 896171
Name: Fantasy, Edith

Sch Crs /Sec	Title	Term	OCT	DEC	MID	APR	ALT	TYP	TR
SAL 2150 105	SOCST 20	12	65	65				R	
SAL 2200 103	MATH 20	12	84	78				R	
SAL 2501 101	ACCTG 20	12	68	55				E	
SAL 2611 101	FD ST 20	12	79	81				E	
SAL 2100 304	ENG 20	34						R	
SAL 2230 302	BIO 20	34						R	
SAL 2240 303	CHEM 20	34						R	
SAL 2302 301	FR 20S	34						R	
SAL 2413 304	CALM 20	34						R	

Figure 20
Course Request Weighting

John strongly supported the concept of including the capability of balancing student work loads between semesters. Having learned some of the issues programmers face in taking ideas through to completed programs, John observed, "One of the problems with trustees and others is their lack of understanding of what is involved" (Personal Interview, April 1992). What appeared to be an easily manageable request that trustees heard from parents, that of unbalanced work loads between semesters, involved a great deal of effort by the programmers.

Individual schools were given control over the assignment of the weighting factors as well as the order of importance to the student scheduling program (Figure 21). In this example, the scheduler is instructed to use the weight as the most important factor for deriving a student schedule, followed by the number of seats available in the class and then balancing between gender as the third criterion. Various thresholds can be set for these balancing variables--here set at 10%. These thresholds determined how much latitude the program is allowed in using the various balancing factors (10% seating in a room containing 30 seats provide a latitude between 27 and 33 seats).

```
CRSSCH-1                    91-92     Test School                     30-Mar-92
                             Student Request Scheduler

 -Balanced-BY-      Main Pass:                  Y    Students selected:     421
  Weight   1 10%    Partial Pass:               Y    Processed:  421 - 100%
  Seat     2 10%    Clear Unlocked Section:     Y
  Sex      3 10%    Exceed Maximum Seat:        N       Success:  398 -  94%
  Term        0%    Balance Across Section:     Y    Conflicted:   23 -   5%
                    Schedule By Difficulty:     Y  Request Fail:   35 -   0%
                  [ F ...  ]
 Messages
Student 003104 has duplicate courses (2100) without sections and no restrictio
Student 922035 has duplicate courses (1230) without sections and no restrictio
Starting main pass...
Starting partial pass...
Updating REQUEST data...
Updating SECTION data...
Scheduling rate 832 stu/min (overall rate 219 stu/min)
```

Figure 21
Scheduler Output

The speed of the scheduler was considered to be important. In the example (Figure 21), the overall processing rate was 219 students/minute on a desktop microcomputer (typical rates were approximately 500 students/minute). The overall rate included the amount of time for the program to read the data from disk and write out the results back out to disk. The scheduler rate was 832 students/minute. The elapsed time for this simulation was just under two minutes (421 students/219 students per minute). The elapsed time for the entire school was approximately six minutes. This compared with the elapsed time of approximately 8 hours with the previous scheduler (i.e., approximately 80 times faster). The result of such dramatic speed increases meant that the school administrators could run timetable simulations *during* the day. In addition, they could run many more simulations within the scheduling period of May and June of each year.

The speed of the newly developed scheduler was greatly increased over that of the previous one. Course weights were devised to permit the scheduler to balance student loads between semesters. Since the new programs were written "in house" and "from scratch," the algorithms and documentation were available for the first time in the district.

The new student scheduler was one of the major components that had to be ready by May 1989. There were many other reports that also had to be completed--course tallies, conflict matrix, section load report, room utilization, teacher utilization, etc. All of these also had to be operational at the same time. Lyle recalled his role in this stage of the project:

> We needed a number of reports that just had to be done. You can't build a timetable without a section load report. Or how about the program to print ID cards. We could have so easily gotten off track. Living with the guys I could say, 'What am I going to tell 1,400 kids that their ID cards aren't done?' Then 'boom' the program was done (Personal Interview, March 1992.

These reports were being concurrently completed by other programmers on the team concurrently while Lyle was completing the scheduling program.

Scheduling Speed Factors.

The significant increases in speed of running timetable simulations resulted in increased use of the scheduler by the schools. While previously it was only possible to run five or six timetable simulations, some schools began to run forty or more simulations to provide the "best timetables" for their students. Lyle recalled: "The dramatic increases in speed could not have taken place without the move to C" (Personal Interview, March 1992). The choice of a computing language other than the traditional COBOL language was instrumental in providing the tools for the programmers to obtain the faster speeds. Lyle made the following observations:

> The speed also came from doing everything in memory--the slowest part is reading the data in and writing it out when you're done. C's memory allocation let us load everything in and free up what was left unused. C allows for a number of things that can't be done in COBOL. When we were first developing o·· standardized routines in late 1988 we wanted to make better use of memory allocation. Some of the first common routines were ones that freed up memory that a program didn't need any more. Because C doesn't care about memory boundaries, we had to develop some memory overflow routines and check sums to make sure we didn't write things in memory beyond our allocated space.
>
> C has constructs that COBOL doesn't have--things like pointers, memory management, bit manipulation, things like that. It has a lack of strong typing, so we could create our own types. In the scheduler take a look at something like the number of periods per day or number of days per week. So everything in the scheduler was put into integer format for speed. An integer compare takes one machine instruction. Comparing a four character field takes a lot of machine instructions.

> In COBOL you need 4 bytes which can hold an integer of 4.3 billion. That much room is taken up all the time. C has long integers, integers (which are fast--usually 32 bits depending on the machine), short that are 16 bits, and character which are 8 bits. Take the number of terms--you only need 1 byte not 4 to hold the maximum number per school. If you wanted to have the scheduler look at the number of terms in a school of 1,700 with about 10 course requests for each student, you're talking 17,000 term items. If each term took 4 bytes instead of 1 that C required, you'd need 4 times as much memory. That piece of information alone saved about 1 meg of memory. Courses and sections would have to take 8 bytes, program restrictions 4 bytes, and so on. You just couldn't fit it all into memory if you used COBOL tables. By using types and doing dynamic memory allocation you can load everything into memory and get the speed you need (Personal Interview, March 1992).

The choice of the computing language made constructs available that did not exist in more traditional commercial computing languages. Two such constructs--memory allocation and types--permitted the large increases in speed realized in the student scheduler. The *technical* choice of the computing language had direct impact on the common education problem of student scheduling. John illustrated the success of the new scheduler when he stated that, "The single biggest thrill of the new SIS is how you can go from one module to another effortlessly. It was so fast you could do a bunch of simulations a day" (Personal Interview, March 1992).

User Input.

John saw his role in explaining to the programmers "...how things worked in a school". This is illustrated in the following recollections:

> When you're in a school you're looking out for what works quickly for kids. This is your key at being 'credible'--you need information at your fingertips. Once the programmers caught on to how people in schools thought, then they began to do things like a school wanted them done. Most of the time now they don't need to ask, but back then it was foreign stuff to them (Personal Interview, April 1992).

John also learned things about how programmers approached problems and some of the difficulties that they faced.

> I got a good understanding of what programmers do and what's involved. Schools still call me about things they want done. I've got an understanding of what needs to be done to make changes in a system. Once you understand that, you can defend the programmers on the change or clarify what is required to the school (Personal Interview, April 1992).

As described earlier, one of John's major concerns with the existing student system and any of the microcomputer based systems which he evaluated was the use of a highly modular approach. When registering a student, a user had to exit from the registration program, enter the course request program and enter requests, leave the course request program and enter the scheduling program, and so on. The entering and exiting of programs introduced a great deal of time for the student waiting for either the counselor or administrator. John pressed for a system with *seamless* transition from one module to another. This suggestion of a seamless transition was one of the design goals that would not have occurred without John's participation as a member of the development team. This transition would facilitate the activities that were occurring at the school during a registration (Figure 22).

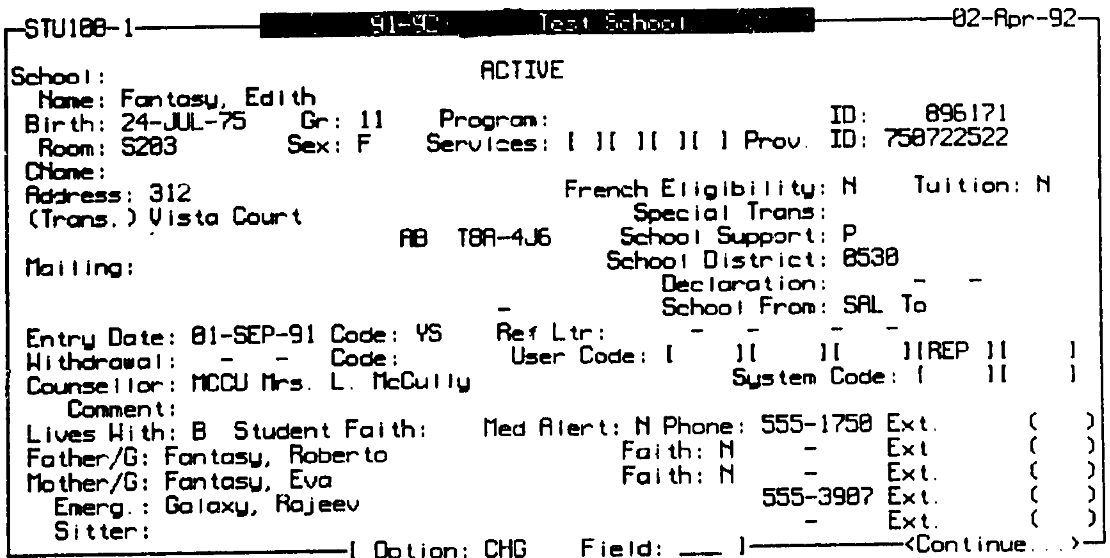

Figure 22
Demographic Module

John defended the need for a user to be able to go directly from the demographic module directly to any other module. He reasoned that a "Macintosh like" menu that "popped up" (Figure 23) would allow an administrator to go to whatever module was needed. His reasoning was that at registration time it was necessary to go to course requests, scheduling, fees, or any other module necessary to deal with a student's registration. It was also necessary when a parent came for an interview in which information was required from the attendance, fees, or other systems.

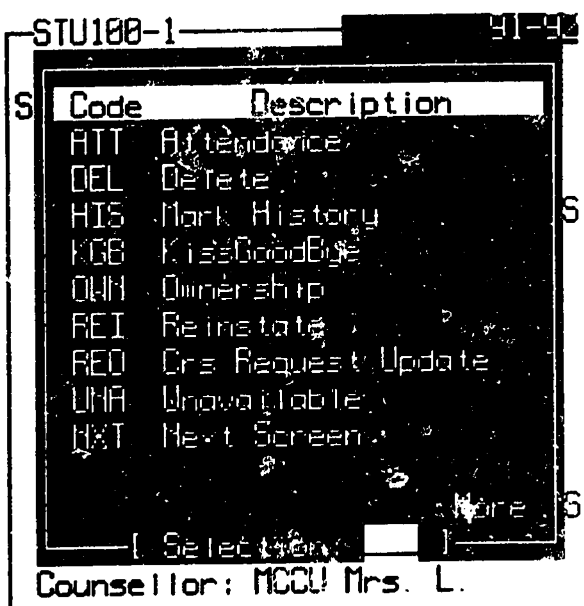

Figure 23
"Pop up" Menu in Registration

Selection of one of the options kept pertinent information on the screen while displaying other desired information (Figure 22). The "seamless" moving between modules was one of John's contributions to the SIS project that reflected how schools actually operated and used the data. Without John's participation, this would likely not have happened since the programmers were under severe time constraints and the creation of such capabilities required significantly more work.

Anecdote

Another of John's contributions to the design of the SIS came in the form of a humorous anecdote. This anecdote had to deal with KGB - "KissGoodBye" (Figure 23). The student record system was built to ensure that students could not be registered more than once ("duplicate students") or that student course requests were not entered more than once ("duplicate courses"). When a registration took place, the program searched the list of legal and common names with similar ages to that of the student being registered. If a potential match was located, the program would prompt the user that a potential registration already existed and that a transfer should take place rather than a new registration being permitted. Data concerning students erroneously registered twice by entering slightly different information during the second registration were removed from the system by being marked with a "deleted flag". This flag was a logical variable that indicated the registration was not an active student. In order to completely eliminate information about the student from the SIS system and not simply have the student data marked "inactive" by this flag, the student and all associated records from the various data files had to be removed. The programmers referred to this procedure as "expunging" the student. John felt that the schools would have difficulties with the term expunge and proposed KGB, "kissed them good-bye" after the Soviet secret police. When a school wanted to truly remove a student record from the system and not to have them simply marked as inactive, they "KGB'd" them.

Unavailable and Course Sections.

The fourth deficiency of the old scheduler was some of the "work arounds" required to use the software in a school situation. In the example of a student who had a part time job outside of the school, specific portions of such a student's timetable had to be protected so that classes would not be scheduled for times when the student would not be at the school. John would create "dummy courses"--courses that did not exist, but which were created for specific time slots during the day. A student that would not be in the school at those various time periods would be given these "dummy courses" as required courses so that the scheduling program would register the student into those courses and would not schedule other courses for that student's time slot.

John discussed this with Lyle and they proposed a new attribute referred to as "unavailable" (Figure 24).

Figure 24
Unavailable Menu

By evoking the option to make certain time slots unavailable to the scheduler, the program dropped to an additional menu (Figure 25) in which up to four time blocks during the year could be "blocked out" from the scheduler. When the scheduling program was building the student's timetable, those times that were unavailable due to work or other reasons would be removed by the program before courses were scheduled in the available times. This innovation was a direct result of an experienced school administrator working with the programmers and greatly improved the day to day efficiencies of the software.

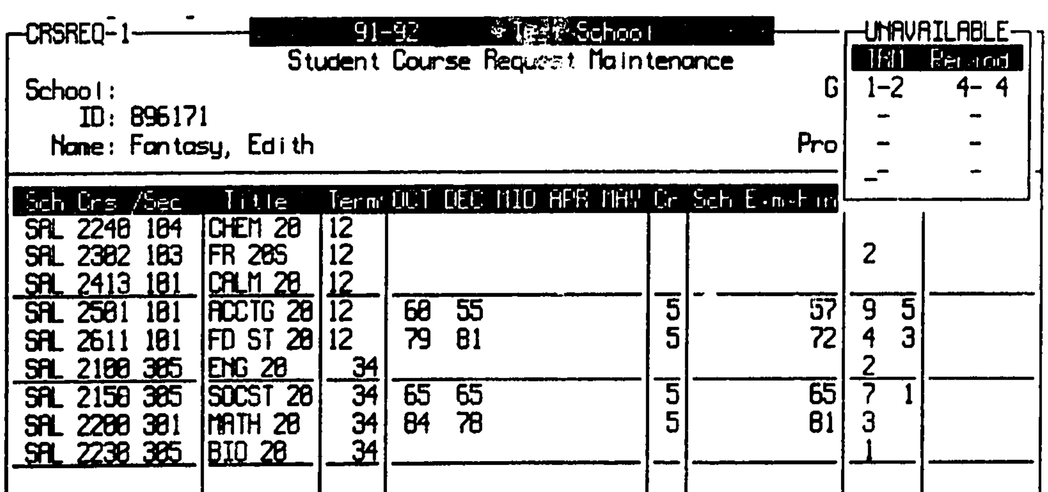

Figure 25
Making a Student "Unavailable"

John's evaluation of the program which set up courses, and sections within courses, was also very positive. Some high schools operated with a mixture of semestered and full year courses. This resulted in some reporting periods during which marks were entered into just the semestered courses, at other times when marks were entered for full year courses, and other times for both. At mark entry time supervision of secretarial staff was required and occasionally marks were entered into report cards in the wrong reporting periods. John summarized the modifications to the course/section software as follows:

> The course/section stuff is a real Cadillac! It has a high level of 'idiot proofness'. When I set up a course/section at the beginning of the year, I set it up with the marking schedules required for it. It gets set up at the beginning and then 'no worries.'
>
> For example, in the old days you would have a 5 credit course with marks in marking periods 1, 2 and 3. A non-semestered course could have marks in marking periods 1, 3, 4, and 6 (our school has 6 marking periods). If you are in the fourth marking period, the program automatically produces sections with marking period 4 in it. The secretaries can't miss anything! The system does it all for them (Personal Interview, April 1992).

These two examples of solutions to "work arounds" were indicative of the many areas of impact between the programmers and users that John had on the project.

District Computing Events

The fall of 1988 saw a number of computing initiatives take place throughout County X. These events occurred concurrently to the events in the case. Two major events took place in administrative computing during the case study period. One major development was the installation of Macintosh based local area networks in each of the senior high schools in the district. This included the installation of Macintosh computers for use by the administrators and counselors, as well as a file server and laser printer for use on the network. Prior to these computers being installed, access to the pupil record system took place primarily through terminals (although some PC's were used as well).

A second event, the introduction of a Macintosh based electronic mail (E-mail) package in February 1989, marked the first *strategic software* acquired by County X that did not run on the mainframe computer. This initiative came through Lloyd's research and development fund and was to serve as a prototype for electronic mail packages intended for use on a mainframe computer. Initially available only to central office and the senior high schools, this software became an immediate success as administrators and schools could communicate in a "user friendly" fashion with a graphic user interface on their own personal computer (they did not have to log into the mainframe for electronic mail).

The ability for educational administrators to send mail, along with attached working spreadsheets and word processing documents, directly to other administrators was a major milestone in the acceptance and utilization of computers by administrators. This software was finally made available to all schools (October, 1991) when local area networks were installed in each elementary school (September, 1991).

The original student system continued to run on the old mainframe computer until October 1989. During the period of this study there were changes in telecommunication speeds, more networks were introduced into the schools, junior high school marks reporting was introduced, etc. During the 1989/90 budget preparation (December 1988 - April 1989) Lloyd's supervisor questioned why various budget initiatives were being presented. Lloyd's response dealt with the need to run software on multiple platforms, the need for machine independence, events taking place in the industry, and so on. To this his supervisor responded, "But where have you stated this?" (Personal Interview, April 1989). As a result of this question Lloyd prepared a position paper for County X (presented November, 1989) outlining the development architecture for administrative computing.

This position paper delineated the direction of computing for the 1990s as well as identified key technologies to be monitored as the 1990s unfolded.

County X acquired its first 9600 baud modem in May 1989 and its first Intel 80486 based PC in August 1989. These two acquisitions of recent technology illustrate the continuation of the trend for early acquisition of evolving technology .

Emergent Events

There were a number of emergent events that arose following the study. Some of these events were organizational--the combination of some departments and the splitting of other ones. Some events were computer related--the expansion of the use of electronic mail and the expansion of the telecommunications infrastructure. Other events saw the implementation of centralized data for electronic access.

Emergent Events

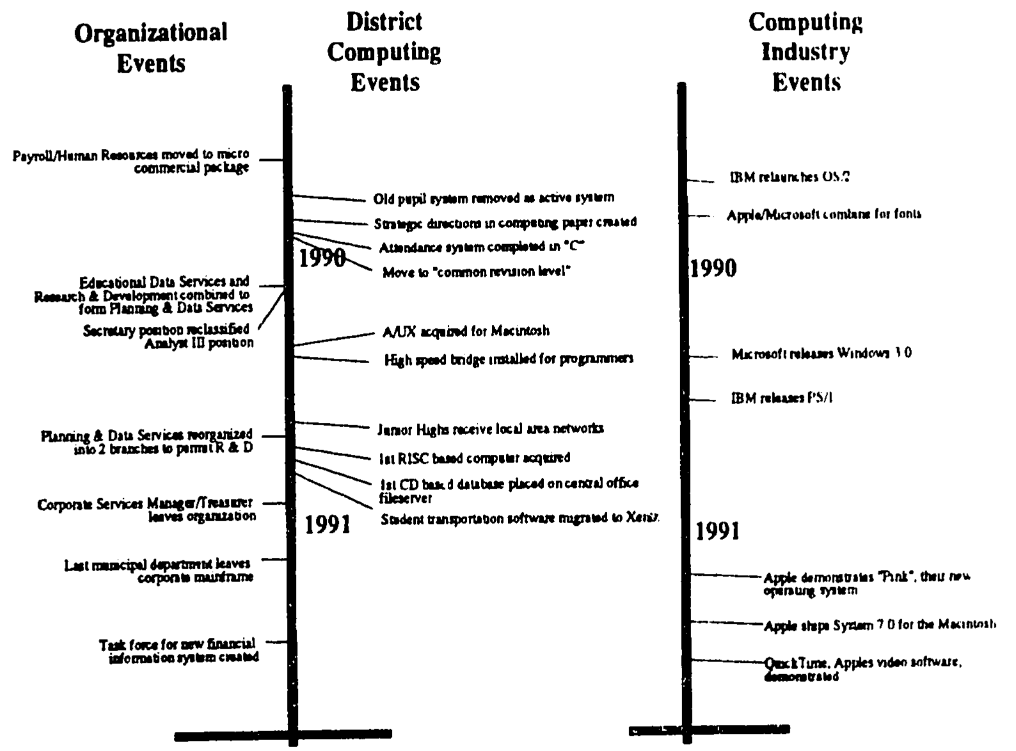

Figure 26
Emergent Events

These events are briefly outlined below:

Departmental Reorganization

Two departments were consolidated following the study period. Lyle's Education Data Services department and Lloyd's Research and Development department were combined in January 1990. The rationale for this consolidation was to make better use of existing resources and ensure that the previously separate domains of administrative computing in the schools (Lloyd's responsibility) and the centralized computing facility (Lyle's responsibility) would continue to evolve in a coordinated manner.

Separate education and municipal finance departments were created in April 1992. Boris, the Treasurer, left the organization effective December 1990. The position of treasurer was assumed by Rick, who continued to maintain the responsibilities through April 1992. Rick and Frank had advertised for a treasurer and interviewed candidates in 1991 but were unable to agree upon a replacement for Boris. This lack of a full time treasurer, combined with high staff turnover in the finance department, resulted in a situation where the Board received untimely and inaccurate reports. In September 1991, the Finance department projected a deficit for the 1990/1991 school year. As a result, steps were taken to reduce expenditures. However, instead of the projected deficit, the year end budget showed a large surplus. This situation precipitated considerable discussion among trustees about the effectiveness of the corporate finance department and helped contribute to the agreement by the Board to separate Education Finance and Municipal Finance.

A financial information system (FIS) task force was established in 1991 to evaluate alternative software for all financial information systems (general ledger, purchasing, payroll, human resources, work orders, etc.). This task force was established by Rick and was chaired by Joe, the municipal computing manager. Initially there were no education representatives on the task force. Later one of the Associate Superintendents was added to the group and was the only education representative. This process was similar to the 1979 situation (Figure 15) when the municipal administrators decided to buy a computer and education participated in a reactive manner.

Joe chaired the task force and Lloyd, now director of computing for education, was not a member but had to make representations through the Associate Superintendent. Joe withdrew from the evaluation of "open systems" software in May 1992 by stating that open solutions did not meet the municipal standards. At the time, final vendors were being selected for financial packages. This left Lloyd and the education finance groups scrambling to evaluate software packages. Again, this was similar to previous events, such as leaving the rewrite of the SIS system to the end of the conversion task list.

Communications Infrastructure

The "communications infrastructure" refers to the equipment required to allow for various Macintosh and PC computers to access either the central computer or various file servers. This equipment consists of modems, bridges, wiring, and so on. The infrastructure was quite simple prior to the study (Figure 27). As of August 1987 there were four 1200-baud dial-in ports for remote users and an additional four dedicated data lines from two senior high schools. All other student information was keyed by data entry personnel from forms sent in from the schools.

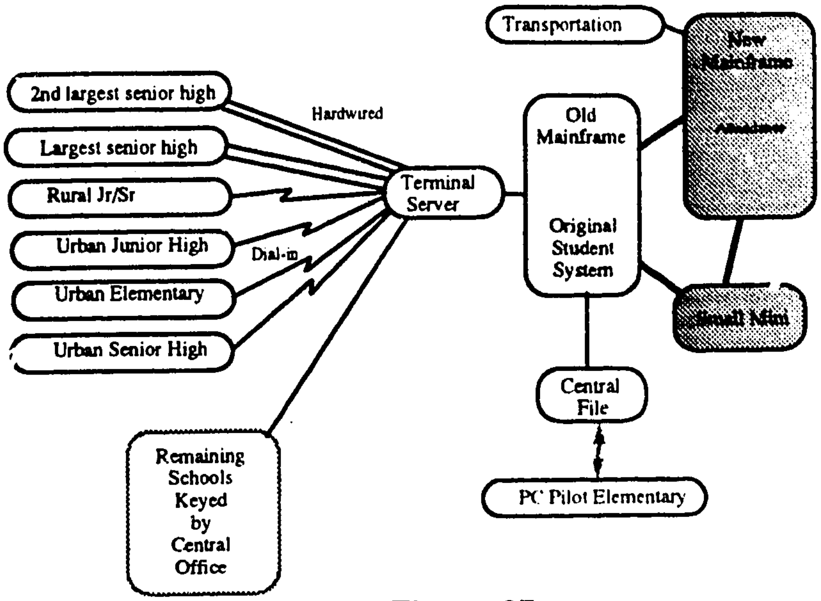

Figure 27
Computer Topology - August 1987

After August 1987, there was a rapid expansion of the number of microcomputers throughout County X. By May 1988, every school was on-line for student demographic information. By September 1991, every school had a local area network consisting of Macintosh computers and a file server. In Central Office the Facilities Department was operating a work order system on a PC based network and Human Resources and Payroll Departments also were operating a PC based network for some of their operations. All of this equipment was interconnected (Figure 28) into a wide area network (WAN). The increase in both numbers of microcomputers and the complexity of the network in a relatively short time is evident. By December 1991, Lloyd's group provided support for over 300 Macintosh and 30 PC microcomputers. This support included acquisition and installation of hardware, software, and associated training.

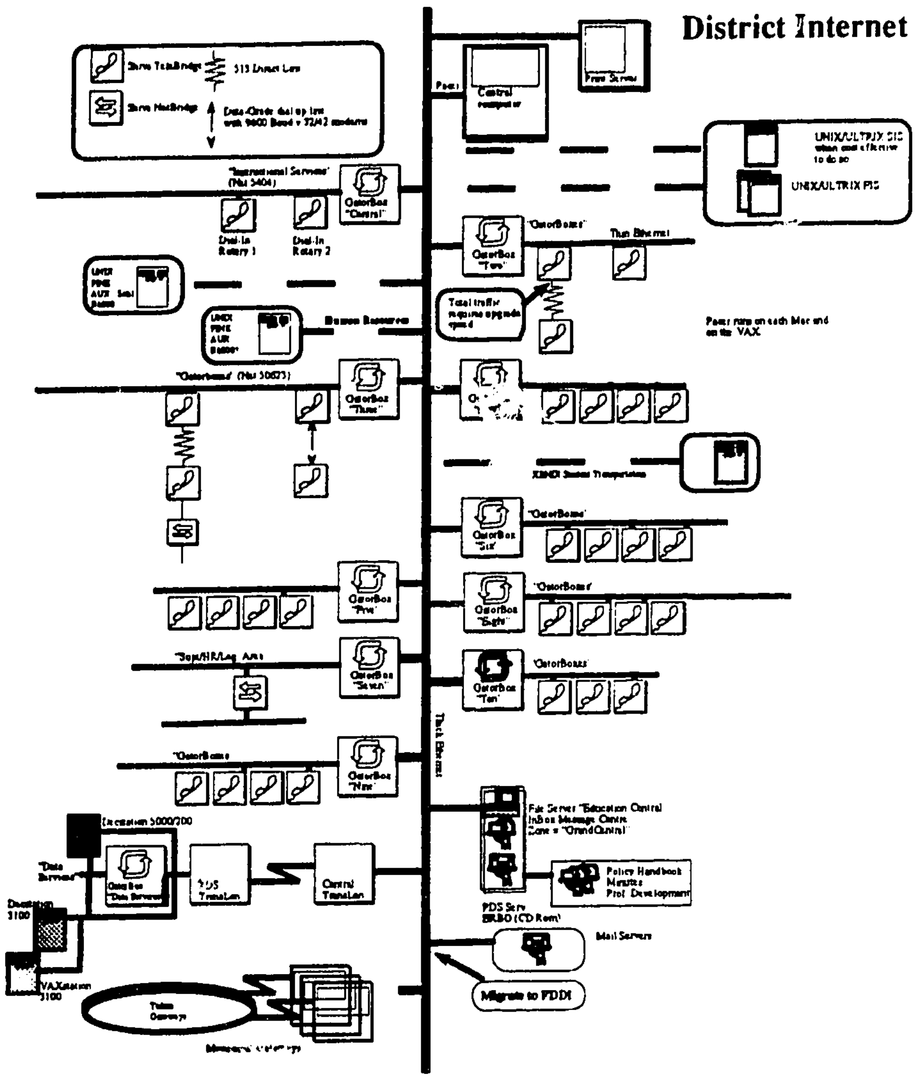

Figure 28
Computer Infrastructure - September 1989

In addition to the increased numbers of computers and complexity of the network came the added tasks of conducting diagnostics on the network and analyzing network performance. These emergent tasks required software tools that were just being developed within the computing industry and required high levels of "discovery learning" on the part of the programmer analysts assigned to maintain the network. The change from August 1987 (Figure 27) to September 1989 (Figure 28) illustrates the *rapid* evolution of the computing environment within County X.

Introduction of electronic mail

Lloyd acquired a microcomputer based electronic mail package (February 1989) which was implemented on a pilot basis during the study. Like other initiatives, electronic mail was introduced as a prototype to examine the issues and implications of new software. As microcomputers expanded, first to the junior high schools and then to all elementary schools, this application expanded as well. The ability of individual administrators to have almost instantaneous communication with other administrators, including the sending of word processing documents and spreadsheets, greatly altered how communication took place within County X. By December 1991, every administrator in the district had access to this electronic mail package.

Other developments

The availability of *RISC* (Reduced Instruction Set Computing) CPUs became more affordable from approximately 1988. Computer manufacturers found that making larger, faster computers was becoming increasingly complex and expensive. They determined that simpler CPUs could be constructed by eliminating unused or under utilized instructions (thus, a reduced instruction set). These RISC CPUs could then be made to run substantially faster than the more complex CPUs and could also be produced more cheaply. The potential for significant cost reductions along with increased capacity was a major criterion used by the SIS development team to establish the move to machine independence. Lloyd acquired the first County owned RISC based computer in 1989 to prototype the SIS migration path.

The reasons Lloyd acquired the RISC based computer were not entirely technical in nature. There was a major financial reason as well. Grants to County X from the provincial government continue to decline. The 1992/1993 structure for provincial grant funding was a three percent increase. Changes to other regulations concerning student counts resulted in the effective grant increase being approximately two percent. The organizational forces to keep instructional programs functioning in an environment of economic restraint implied that operational departments such as Lloyd's would increasingly be under pressure to reduce operating costs. Lloyd's department acquired a RISC based computer in order to examine the possible move to machine independence and to evaluate the impact of this technology on the district's long term "cost of ownership" of computers.

Other innovations took place. The first CD-ROM based application, a research data base, was added to the central office local area network and made available to all Macintosh users in August, 1990. This event, along with the replacement of County X's policy handbook with an electronic one (September, 1991), illustrated the move of applications interconnected on various computers. This multi-vendor based model was outlined in the strategic directions paper.

Summary

The organizational structure of County X was presented along with an overview of Phase II of the SIS project. Three timelines were then presented--events antecedent to the study, events within the study, and emergent events. Each of these timelines illustrated three threads--organizational events, computing related events within the district, and computing events outside of the district. Computing and organizational events related to the study that emerged after the study period were briefly noted.

The antecedent events timelines included discussion of antecedent technology, a milestone event (the termination of a computing family by a vendor), decisions that took place prior to the study period, developments that had an impact on the SIS system, and trends in technology prior to the study.

The case study itself had a number of major elements: the SIS project tasks, the relocation of the programming team to a new location, the choice of programming language for the project and its implications, the adoption of a philosophy of "machine independence", the student scheduler, the role of the boundary spanner, the formation of separate education and municipal computing groups, and the evolution of computing and networks within County X.

Various emergent events within the district, both those related to the organization as well as computing, were presented. These included the combining of Lloyd's and Lyle's departments into a single department, as well as the eventual separation of finance into separate municipal and education departments. The large increase in numbers of computers and complexity of the network in the district were also noted.

Chapter 5

Analysis and Results

The problem stated at the outset of the thesis was, "What is required for the successful design and implementation of a computer based student information system in a school district?" In addition to this major problem a number of sub-problems were also stated. The sub-problems will be examined first before returning to the major problem.

Key Events

The first sub-problem was, "What were the key events in the design and implementation of this system?" Various examples of these events were cited: "Which programming language should be used?"; "Should the software be developed in-house or should a commercial package be acquired?"; "If the software is developed in-house, should new development techniques, such as Martin's concept of prototyping, be used?"; "Could the design be improved by utilizing a person in a boundary spanning role?"; and "What data elements should be used?"

The answer to the first sub-problem, ""Which programming language should be used?," was based on the antecedent events to the study. The milestone event, when the computer manufacturer decided to phase out a computer family, had a far reaching impact on the design of the student information system. Both the decision to move to "machine independence" in design and the decision to utilize the C language both were a result of the computer vendor's decision. Had this event not occurred it would have been unlikely that the radical changes in design of the data files and programming language would have been made. Other design events included the move to a data base manager. This decision allowed for the creation of static and dynamic records, which in turn solved the administrative problem of students attending more than one school concurrently.

Another key event was the relocation of the SIS project team out of the central office administrators' building to a school location and Frank's decision to split the corporate computing group into separate education and municipal groups. Without the relocation of the SIS team out of central office to a school, the continual reallocation of priorities for the programmers likely would have continued and the project would not have met the timeline targets. Without the formation of a separate education computing group it was highly probable that many of the SIS programmers would have left the district for alternate employment.

The second sub-problem was "Should the software be developed in-house or should a commercial package acquired?" John's presence facilitated this decision by representing the interests of the schools on the project team. John was delegated the task of evaluating microcomputer based software solutions. When he decided that none of the packages evaluated gave as much functionality as the schools were already utilizing, he recommended a rewrite of the current software. His personal credibility with the Superintendent's Team, as well as the school principals, resulted in widespread support for the decision.

The third sub-problem was, "If the software is developed in-house, should new development techniques, such as Martin's concept of prototyping, be used?" At the beginning of the project the normal development methodology was to define requirements in great detail using techniques similar to those outlined by Carlsen and Lewis (1979).

These highly structured techniques required a great deal of analysis and user acceptance of definitions prior to the beginning of programming.

The short timelines of the SIS project did not permit such an elaborate methodology. John's total participation allowed Lyle and his team to develop prototype code for immediate feedback. Since John was empowered to make decisions on behalf of the schools, there was very little time lapse between the construction of a prototype and its revision and completion. Where John was unsure of how his colleagues would react, he was able to convene meetings on short notice to obtain consensus. The prototyping method was able to demonstrate to the users in an effective way what the programs would do and how they would perform. This immediate feedback to the users would not have been possible under the more standard development methodologies like those outlined by Carlsen and Lewis. The use of the Martin style prototyping was highly successful in the project.

The final two sub-problems--"Could the design be improved by utilizing a person in a boundary spanning role?" and "What data elements should be used?"--are linked. As just noted, John was able to evaluate prototypes in very short time frames. His role as a boundary spanner, primarily between the programmers and the schools but also between his school administrators and the superintendent's team, permitted decisions to be made quickly. In this capacity the development times were reduced as compared to how they would have been had John not been part of the project.

John's participation had a great impact on Lyle's decision to move from the traditional "one student - one record" file structure to a single "static" record and variable numbers of "dynamic" records. John often stated, "This is how it works in the schools." John indicated that he had to use duplicate students in order to "work around" situations such as a student being in two schools at the same time. As John described situations such as this, Lyle was able to conceptualize the file definitions and data elements in those files differently than he had done previously. John's presence on the development team led to the use of "Macintosh like" pop up menus, the ease of switching between different modules within the SIS system, and the user orientation of the software to "how it was done in the schools".

Organizational Issues

The second sub-problem was "What were the key organizational issues?" Examples of this included "Who were the key decision makers and influential participants in the design and implementation?" and "Was the organizational structure appropriate?" Each of these sub-problems is addressed below.

Organizational Issues and Structure

I concluded that the county form of government as it existed at the time of the study was a major impediment to computer development in the district and constituted the key organizational issue--the structure of governance was inappropriate. The county form of government was comprised elected of officials that were municipal as well as from two urban areas (Figure 11). This fostered an environment of competition. The county form of government made it difficult for corporate employees to meet different management priorities.

As described earlier, the *dominant* municipal management style resembled Mintzberg's traditional bureaucratic while the *dominant* education management style resembled Mintzberg's professional style. The "way business was conducted"--how decisions are arrived at, what priorities are set, and how these are communicated, are different in these two management styles. While administrators within organizations often use many different management styles (Hersey & Blanchard), the dominant styles will cause conflict when viewed over a longer period of time. Counties or school districts with such competing management styles will create an environment that is very difficult to develop computer applications.

Johns (1988) described the vertical division of labor (the apportioning authority for planning and decision making) and the horizontal division of labor (the grouping of basic tasks to be performed into jobs and then into departments so that an organization could achieve its goals). "It is entirely possible to conceive of a firm or institution that has well-motivated individual members and properly led groups and still fails to fulfill its potential because of the way their efforts are divided and coordinated" (Johns, 1988, p. 502). The SIS group in County X contained highly motivated individuals, but individuals were often redesignated to other tasks. This situation became critical enough to relocate the team to a location outside of the administration building.

The SIS team was a highly cohesive and highly motivated group with a development philosophy more closely aligned with the educational component of County X than with the municipal component. Mintzberg (1989) noted that the *ideology* or culture encompasses the traditions and beliefs of an organization. He noted that within an organization the full-time employees were influencers and formed an *internal coalition* (p. 98). Mintzberg (pp. 106ff) further described four "situational" factors that influence the decision making system of an organization. The increased automation technical system, including the knowledge base of the system, moves an organization away from a bureaucratic style of management towards an organic style. The knowledge base of the SIS team was well advanced when compared to the understanding of information systems by the municipal management. The tendency of the SIS team was to continue the *dynamic* organic approach initiated by the first data processing manager but significantly altered by the arrival of Boris and Joe.

Mintzberg (pp. 108,109) described another situational factor of the "environment". The "environment" referred to characteristics outside the organization's context (e.g., market, political climate, economic conditions, etc.). Mintzberg described some of the factors at work: the more dynamic an organization's environment, the more organic its structure; the more complex an organization's environment, the more decentralized the structure; and extreme hostility in its environment drives any organization to temporarily centralize its structure. For the SIS team, the environment was very dynamic with tremendous changes taking place in hardware and software technology. This led to increased tendencies within the team toward organic decision making. The high level of complexity of application development also increased the tendency toward organic decision making. This tendency was opposed by the Boris' centralized style of management.

Bluhm (1987) also described two different organizational stages dealing with computer development (Figure 5). These stages were the expansion stage and the mature stage. The mature stage was characterized by the proliferation of broader and more advanced applications and an increased number of personnel becoming more specialized in the use and programming of the computer. The expansion phase was a period of contagious, unplanned growth, characterized by growing responsibilities for the EDP director with loose decentralized organization of the EDP facility coupled with few explicit means of setting project priorities or crystallizing plans (p. 31). Lyle's SIS team was placed in the style suitable for the Bluhm's "mature stage". The SIS project, with its tight timelines, was better suited to Bluhm's "expansion stage" where prototyping, ad hocracy, and a less structured environment were required.

Without examining individuals' management styles, there existed within the structure of County X two differing management styles and hence the mechanisms for decision making differed. Mintzberg (pp. 110ff) described different management styles or organizations. The SIS team largely reflected the school district's 'professional' organization. The professional organization differed from a machine bureaucracy by emphasizing the authority of a professional nature--the power of expertise. Lyle had very high levels of technical expertise. John had high levels of school administrative expertise and computerized student record expertise. Lyle's technical expertise complemented John's school based expertise. These two individuals integrated with the professional style of decision making characterized by education and did not integrate with the highly structured styles of municipal administration.

Key decision makers

Many individuals were involved with the different kinds of decision making within the study. Within the SIS project team Lyle was th major decision maker. Lyle played a key role with respect to decisions concerni e computing language used for development; he played the leadership role in fending the SIS project with the Superintendent's team, and he defined the internal technical standards for file layouts, library routines, and other design issues.

John and Lloyd were the two individual boundary spanner decision makers. Each occupied a different boundary spanner role (Figure 9). John was a liaison between both the SIS development team and schools as well as the SIS team and Frank's management team. By having John represent the schools' needs to the SIS project team, the usability of the software was improved for a school setting. John was pivotal in representing to the programmers "how it was done in the 'real' world". John demonstrated prototypes of various screen displays to selected school clients and presented their feedback to the programmers.

Lloyd 'spanned the boundaries' between the SIS development group and Frank's management team in a more formal fashion than did John. John served as an informal 'checking and verifying' agent--an independent check of events. Lloyd represented the project in task assignment, task reporting, budgeting of resources, and the establishment and defense of priorities for the project. Within the SIS group, Lloyd served as a "sounding board" for Lyle in computer architecture, language, and other technical issues that affected the project but were difficult to articulate to senior management.

Organizational summary

There were several factors in the organization that had the potential to impede the success of the SIS project. One of these factors was the attempted use of a Bluhm style "mature structure stage" when an "expansion structure stage" (Figure 5) was required. Another factor was the different management styles between the professionally oriented education group and the mechanical bureaucracy of the municipal group. Both of these factors were addressed by first moving the SIS team out of the central administration building and then in the formal splitting of the SIS team into two separate computing groups. This splitting of the SIS team permitted a rapid change of the project participants to the Bluhm style expansion stage organizational structure.

The move to an organic style of management was important to the eventual successful completion of tasks within the timelines. Management requires feedback in this style of decision making and that feedback was furnished by John and Lloyd's boundary roles. Their separate and independent feedback gave assurance to Frank and his team to permit less structured objectives and reporting techniques.

Change Model

The third sub-problem was "Did Fullan's (1982) change model adequately account for the changes in the County creating and implementing a computerized decision support system?" Fullan noted that change was a *process*, not an *event* (p. 41). The SIS project supported Fullan's point. "The SIS Project" was not a specific event but was a series of events in a process. This process underwent the phases outlined by Fullan--initiation through implementation to routinization.

Furthermore, Fullan also noted that educational adoption does not occur without an advocate (p. 45) and that one of the most powerful advocates was the chief district administrator along with his/her staff. To Fullan, *advocacy* and initiation of change were key to the successful adoption of innovative programs. Frank's support to move Lyle's team out of the administration building, his support in the splitting of the computer operation, his support for the secondment of John to the project, his support of Lloyd's budget proposals, and priorities clearly indicate support for the concept that the advocacy role is necessary to initiate change in an organization.

John, in his role as representative of the users in the schools, demonstrated Fullan's linking agent role (p. 47). John ensured that the schools' needs were represented among the project developers. His demonstration of prototypes at various stages of the project not only provided valuable feedback to the developers, but ensured a high level of confidence by one of the major stakeholders, the users in the schools, that their concerns were being addressed. This support was crucial in Frank's support in the relocation of the SIS group, as well as the organizationally traumatic event of splitting a department into two different groups.

Fullan (pp. 91,92) noted ten assumptions basic to a successful approach to educational change. The applicability of some of these assumptions can be identified in this case study. Assumption 3 stated "Assume that conflict and disagreement are not only inevitable but fundamental to successful change" (p. 91). Conflict and disagreements were noted in a number of areas: choice of computer languages, differing management styles between municipal and education personnel, the desire by Lyle's group to use a Martin style prototype development methodology instead of a highly structured methodology, etc. In retrospect Lloyd, Lyle, and John sought consensus and based their decision making around this approach. They would likely have been "better off" had they understood that conflict was both inevitable and fundamental to such a process. Knowing this could have eased tensions on the language issue and prototyping issue. It would not have alleviated the leadership - follower mis-match between Boris and Joe on the management side and Lyle and the programmers on the employee side.

The first assumption, "Do not assume that your version of what the change should be is the best one that should or could be implemented", in retrospect, is highly applicable to the people involved in this study. People involved in the change process should examine this assumption and apply it to themselves. For example, throughout the study various examples of "us vs. them" were noted. Once such an orientation takes place, it is very difficult for the participants not to think that "their" solutions are the correct ones and the solutions of "the others" are incorrect. If Fullan's first assumption was periodically brought to the participants involved, it might have helped reduce the position of "defending one's turf" in the change process. Knowing that "change is a frustrating, discouraging business" (p. 92) may sometimes help individuals to bear the change.

The SIS project reflected some of the implementation factors identified by Fullan (pp. 56ff). Category A, characteristics of the change, were attributes of the change itself. This was represented by the 'need and relevance of the change' (Factor 1), by 'complexity' (Factor 3), and quality and practicality of program (Factor 4).

Categories B and C consisted of characteristics of the district and of schools. Category B, characteristics at the district level, was represented by a number of factors. Evidence for the relevance of factor 5, the history of innovative attempts, was widespread within the district in curriculum matters. In addition to curriculum innovations, the early adoption of technology, illustrated in the timelines, demonstrated the history of innovative attempts in computing. Factor 7, central administrative support and involvement, was represented by Frank's support for the formation of a separate education computing department, by his support of John's secondment in the budget, and a number of other instances. Category C, characteristics at the school level, was represented by John's secondment to the SIS project team and his boundary spanning role with both the schools and with his informal interface with Frank's management team. Category D, characteristics external to the local system, was not applicable to this case study.

Fullan cautioned if one or more factors were working against an implementation, the process would be less effective. While some of these implementation factors were not present in the study, I do not think that any of the missing ones were working against the implementation of the SIS.

Fullan noted the single most powerful internal factor influencing continued change was staff and administrative turnover (pp. 77ff). Since effective change involves interaction among users, the removal of key users weakens the conditions that would incorporate or generate new members. The SIS project did not experience any key staff turnover during the time of the study. The impact of losing a key member could not be determined, but the absence of staff turnover likely contributed to the timely completion of ambitious timelines and ensured acceptance by the target users.

Attitudes toward an innovation are concerned with the perceptions of the strengths and weaknesses of the change. The case study time period did not permit the measuring of acceptance by the schools and other stakeholders in the SIS project. Continued current use of the SIS system by all levels of administrators indicate that the implementation process was successful and that the change process towards routinization was underway.

Fullan summary

Fullan outlined a change *process* dealing with change in educational environments. This process is continuous and not a single event. There are three stages to the process: initiation, implementation, and routinization. Fullan's model described the SIS initiation and implementation as illustrated by the relevance of various factors previously outlined. Fullan's model described the change processes outlined in the integrated model (Figure 9). The case study time window did not extend long enough to confirm the final stage of change, that of routinization, but based upon continued involvement with the project, I believe that the final stage would have been represented as well.

Issues and Themes

The fourth sub-problem was "What significant issues and themes emerge from this research that are of relevance to the creation and implementation of computerized systems in educational settings?" These issues and themes are those perceived by the author to be of import and/or use to educational institutions planning or developing information systems.

Prototyping

A prototype in an information system is a partially working program, screen display, or data base that can be demonstrated to targeted users of the system. DeMarco and Lister (1987, pp. 8ff) proposed *iterative design* to establish an environment where analysts and users could discuss and react to working models of various portions of the software being developed. The analyst would make changes to this working model, get further reaction from the users, and so on. This would be repeated until agreement was reached regarding the functionality of the software from the users' perspective. This process was in contrast to the highly structured but still prevalent methodologies requiring detailed design and signoff such as those proposed by Carlsen and Lewis (1979). Martin (1984) described the creation of a working model or prototype for end user feedback as being an important development paradigm (justification). Martin envisioned a systems analyst working directly with an end user to create data base queries, reports, and screen displays that are demonstrated to the end user. The analyst would then discuss the prototype with the user and obtain feedback. The working model created by this process becomes the focus of a debate ensuring that both the user and analyst were talking about the same thing. This on-going dialogue is continued until the users are satisfied that the material meets their perceived needs.

This technique was very important in keeping the timelines of the project within projected limits. Shortened timelines often translate into reduced project costs, but a cost analysis was not performed as part of the study. Lyle and his staff created a number of prototypes--the screens to connect different modules (e.g., demographics directly into history) without "leaving one module and entering another"; displaying performance statistics on the scheduler screen, etc. John met with the programmers and reacted to the prototype's applicability to a school user, how well it met the operational needs of a school user, and so on. Where further clarification was required by other users, John acted in his boundary spanning role by taking the working prototype out to various schools and departments to clarify issues. The positive feedback by users generated by this process generated enthusiasm for the project by both the users and the programmers.

Machine Independence

Computer based applications are now an integral part of almost every school district. Within administrative computing of school districts, the most common form of applications are financial applications (e.g., general ledger, payroll, etc.) and student records. These core applications are critical to the daily operations of the districts. As the hardware section of the timelines (Figure 15) for this project illustrate, computing is changing at a very high rate. This high rate of change has occurred, and is occurring, in a context of declining grants from the provincial government, along with increasing per student costs for programs. Alberta Education (1992) data show that the per student costs rose from $4,458 in 1980-1981, to $5,336 (in 1990 constant dollars). At the same time, the proportion of provincial grants to total cost of education declined from 68.79% to 59.38%.

With a high rate of change in technology in an environment of increasing costs and declining revenues, school districts require strategies to minimize both the costs of software development and the long term cost of ownership of these systems. This project illustrated the potential use of prototyping to reduce the cost of software development. Machine independence, as a stated district strategy, allows for software developed by the district to be executed on machines from a variety of different vendors. Since there is a great deal of competition in the "open software" arena, hardware prices and software license costs are likely to be lower than would be available in a highly proprietary environment.

Lloyd and Lyle predicted that the use of a strategy of machine independence was the best approach to developing computing applications. The cost effectiveness of this strategy was borne out by events that took place after the study was completed. In 1992, the municipal and education components of County X jointly solicited hardware proposals for new financial information system. The costs of the hardware varied, with a high cost of $638,624 for a proprietary hardware solution, and a range of $139,057 to $202,878 for ... different hardware systems running different versions of the same operating system (confidential internal document, 1992).

Lloyd and Lyle's strategy was more inclusive than simply focusing on mainframe hardware. Microcomputer software (e.g., word processing, spreadsheets, etc.) were selected based on the ability for software to be run on a variety of different microcomputers. The rationale was that it takes a large investment to acquire software, install the software, and train the users to use this software. Over a period of time, administrators and support staff develop a great number of files with data critical to their own operations. In the event of hardware changes, due to changing technology or vendor insolvency, the investment costs of acquisition, installation, and training would have to be incurred again. If the software already ran on other hardware platforms, these "start up" costs could possibly be avoided. Lloyd and Lyle were heavily influenced in this approach by the previous experience of a computer vendor choosing to discontinue a specific line of computers.

Network evolution

There was a rapid evolution of the network topology prior, during, and after the study period (Figures 27, 28). The network structure was relatively simple in 1987 (Figure 27) but had changed and become more complex as the various schools in the district were added to the network. These additional schools were connected to the host computer for student records and also were attached to the network as microcomputers instead of terminals. These microcomputers made use of electronic mail and were used to transfer files on the network. The rapid changes in the network topology were accompanied by changes in telecommunications hardware, which permitted higher speeds for accessing the data.

The dependency of the district on its "electronic highway" (the term that was beginning to be used for the network) was illustrated by comments made by Lloyd, "It seems that people are inconvenienced when the mainframe goes down and no one can access the student record system. When the fileserver goes down or e-mail can't run, I almost get lynched coming down the hall" (Personal Interview, April 1992). Lloyd's comments indicate the rapid dependency of district personnel on both the use of electronic mail and the network connecting the various district microcomputers together.

Networks, both local area networks and wide area networks, became a key strategic area of the district's communications infrastructure. The availability and reliability of these networks emerged as being a significant area of support that was rapidly becoming as crucial as was support of traditional applications software.

Certificated vs. classified

All Alberta school districts are accountable to Alberta Education for costs in various summary categories. One of these summary categories is the cost of salaries. The salaries must be broken into certificated salaries (those personnel who are employed as teachers under the teachers' collective bargaining unit) and classified salaries (those personnel who are not employed as teachers). It was noted that school districts resemble Mintzberg's (1989) professional model of organization (Figure 3). In this model certificated staff are the professional component with the classified staff being found in the support component. All of the development programming staff in the study group were in the support component of the model.

The programmers in this project were a highly professionally trained group with most of the members possessing at least one university degree. For example, Lyle had previously been a classroom teacher before making a career decision to enter computing as a change in profession.

During the budget process, Lyle noted, "It seems like support staff are second class citizens. When teachers get a salary increase, we get less than they do. It's hard to get treated as a professional around here" (Personal Interview, April 1992). This feeling was noted, not only within the programmers in the district, but by programmers in a large urban district (Personal Interview, January 1991), by programmers in a large oil company (Personal Interview, July 1992) and by programmers in a large gas transmission company (Personal Interview, October 1992). In the case of the oil company and gas transmission company, the dominant professional group were engineers and not teachers. In these cases there were strong feelings that, unless there was membership in the controlling professional group, the programming staff were clearly not as important as the professional group.

These feelings are noted here in the emerging events section of this thesis since computing is becoming more crucial to educational environments and staff feelings of alienation can lead to costly staff turnover. In development environments which are highly organic, due to the rapid changes taking place, staff turnover can severely impact computer development within an organization (Fullan, 1982, pp. 77ff).

Organizational issues summary

A number of emergent themes have been described that may be of use to districts which are developing computerized applications. The themes noted included the use of prototyping as a methodology in improving the usefulness of systems design as well as potentially reducing the development costs. The strategy of striving for machine independence was described as a potential solution to the rapidly changing environment of computer technology, as well as potentially providing a way to reduce the long term costs of computing. The rapid evolution of the "electronic highway" was noted as school and department users were added to the computer network in order to access the student record system. The rapid dependence of the users on this network was also noted. The last trend noted, described the feelings of programmers within the district who felt that they were not as significant as participants as was the dominant group (certificated staff) within the district. These trends are areas for further study outside this thesis.

Thesis Problem

The various sub-problems of the case study have been described. The major research problem was "What is required for the successful design and implementation of a computer based student information system in a school district?" An integrated model

(Figure 9) was proposed to illustrate the process for the design and implementation of a computer based student information system.

In the integrated model three major groups were noted: the district which existed within an environment which was controlled by an elected board of trustees and regulated by a provincial department of education; a software development group comprised of individuals trained in computing technology, but who were not part of the certificated administrative group; and the educational administrators consisting primarily of the superintendent's team of senior administrators and school principals. Within the group of software developers, different change processes were at work. Fullan's (1982) change model was used to describe the process of implementing change in educational settings within the administrators' group. Martin's (1984) model of information engineering and prototyping was used to describe the process within the computer development group; whereas Tushman's (1977) model of boundary spanner was used to illustrate the interaction between the various components.

The discussion of the second sub-problem, dealing with key organizational issues, key decision makers and the organizational structure, concluded that the county form of government was a major impediment to computer development within the district. Boris maintained a highly bureaucratic management style. This style included a computer development style typified by traditional computer development as illustrated by Carlsen & Lewis (1979). This management style was different from both the professional style, outlined by Mintzberg (1989), and that required in an expansion phase as outlined by Bluhm (1987). My integrated model (Figure 9) did not adequately describe the major interactions between the municipal and education components within County X. The model was over simplified in that it ignored the major stakeholders such as the schools and the provincial education department.

Fullan's (1982) change model depicted the process of change within the educational administrators' component. As outlined in the third sub-problem, the project went through the dynamic phases of initiation, implementation, and into continuation. The timelines of the project did not allow for the examination of the outcome phase of the Fullan model. One of the mandates of senior administrators (Johns, 1988) is the formation and reallocation of individuals into a horizontal division of labor (the grouping of basic tasks to be performed into jobs and then into departments so that an organization could achieve its goals).

One of the requirements for successful project design and implementation is the appropriate organizational structure. Lyle led the new department of Educational Data Services in January 1989. This allowed Lyle and the developers to use the Martin model of development and also the freedom needed to focus on educational initiatives. Frank's advocacy role in the formation of this department, as well as the continued support of the project, was crucial to its completion. Frank's "deferral" of more traditional project reporting allowed for the organic decision making within the project team, as well as the dynamic model (Bluhm) for rapidly changing environments.

Another requirement was the use of Martin's prototyping (1984). This is similar to DeMarco and Lister's (1987) iterative design. John influenced the design of various aspects of the SIS system. One such influence was the balancing of student loads between semesters (Figure 21). One of the key reasons for rejecting a commercial microcomputer based student package was the poor student schedulers and the requirement to exit one module of the software in order to execute another module of the software. John represented these concerns to Lyle and the programmers and this had a significant impact on the functionality and flow of the software. John also took prototypes of the design of the software and described their operations at meetings with various principals. This resulted in an improved understanding of the project by the principals and their positive support helped the programmers to continue working along those directions within days of devising the methodologies.

Another requirement was the role of boundary spanner (Tushman, 1977) or change agent (Fullan, 1982). This role was served by both John and Lloyd in different areas of the organization. The interaction between the schools and the computer development group was primarily performed by John. Being a certificated teacher and highly respected senior high school assistant principal, John possessed credibility with other schools not only on school related matters but in his role as a leading user of technology. This allowed John to bring the various prototypes created by the programmers to the principals and provide the principals' feedback to the developers.

Lloyd served as a boundary spanner in a different way. As project leader, Lloyd primarily interacted with the superintendent's team on issues of objectives, budget, and progress. Lloyd "lived in both the administrators' world and the developer's world". Lloyd's technical literacy allowed him to represent to the superintendent's team issues which were identified by the development group. His certificated administrative background allowed him to represent issues to the development group; his experience also served to create an environment where the programmers could work with a minimum of "political interference". DeMarco and Lister (1987, p. 4) noted that "politics" included communication problems, staffing problems, disenchantment with the boss or with the client, lack of motivation, and high turnover and was more correctly the "sociology" of the project. Lloyd served as a "buffering agent" between administrators and the programmers to allow the programmers to focus on the projects outlined in a very tight timeline.

Lloyd's technical background enabled him to understand and support Lyle's suggestion for the selection of C as the programming language, to support the concept of machine independence for the SIS, establish machine independence for microcomputer software, and to foster the prototyping of electronic mail and the early implementation of hardware and software. Lloyd's membership in the "certificated" group improved the interaction with the dominant professional group within the district.

John was instrumental in directly impacting the software under development. By representing the users within the development group, John ensured that school users would be able to move between modules of the software; that the scheduling module would attempt to balance student work loads between semesters; that the SIS would ensure correct course codes for transmission to Alberta Education through the establishment of course master tables within the district while still allowing individual school courses; created the "kiss good bye" (KGB) for the permanent removal of student demographic data, as well as all associated data; and established the requirement for faster speeds for the student scheduler timetable simulations.

The changes to the integrated model resulting from this study are outlined in a modified integrated model (Figure 29). These changes reflect the important influence of the municipal component of County X as well as the significant roles of the schools and provincial department of education. There were significant interactions between various components of the model. Various boundary spanner roles were not described in the study but are illustrated in the model. These include the roles between Joe's team and the municipal management as well as to the schools (this was indirectly referenced with John's role of confirming the events of the SIS project) as well as Alberta Education.

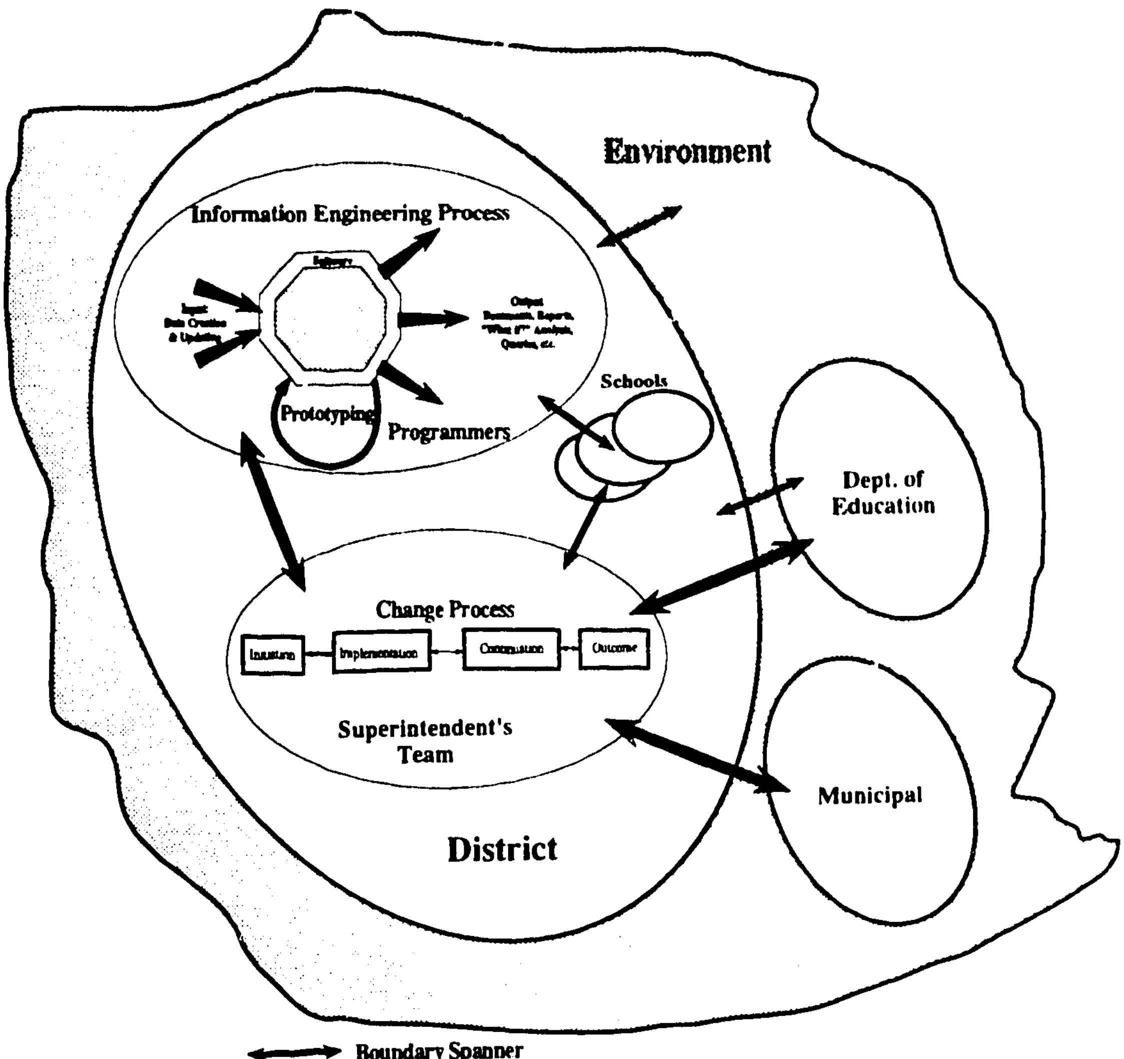

Figure 29
Modified Integrated Model

Another requirement for the successful design and implementation is the availability of highly competent people. It can be safely assumed that programmers will be competent in their particular area of specialization. What is required in programmers for system design are the *analyst* aspects of forward thinking, anticipation of future problems (e.g., new technologies, declining financial revenues, new development methodologies, etc.) as well as *flexibility*. Flexibility refers to the ability to be able to change and adapt in a rapidly changing environment. The impact of many new technological innovations have been described throughout this thesis. More rapid and fundamental innovations will continue to appear. Programmers will be required to constantly rethink their approaches to developing systems as tools such as "object oriented programming", "client/server", and other fundamental changes become more widespread. Administrators must create an environment that attracts individuals with these characteristics and also retain them within the organization.

Chapter 6

Recommendations and Suggestions for Further Study

A number of years have passed since the time period of the study; various events have occurred and many decisions have been made. From the organizational point of view, separate financial departments, education and municipal, were created in June 1992. Each of these departments has its own treasurer. Much of the impetus to have two different treasurers resulted from the earlier decision to have separate computer solutions for financial information. The Board of Education chose to continue looking for software that ran in the "open environment", while the County Council chose to look for software that ran on proprietary hardware chosen for specific municipal solutions.

The predicted cost savings for the use of open systems was supported in the acquisition of two financial systems. The software costs for the two packages were very similar. However, the hardware and maintenance costs provided quite a different picture, especially when examined over five years. The difference between these two solutions was approximately $1.4 million over a five year period (Confidential Internal Documents, June 1992). In addition to these costs, at the end of the five years there would be an outstanding municipal debenture of $908,000 compared to an outstanding education debenture of $289,000. The total difference in five year costs, including the outstanding debt, is projected to be over $2 million. The move to "open systems" clearly had significant cost savings for the district as compared to a proprietary hardware approach to computing.

The Network

The growth in numbers of computers and complexity of the computer network within County X were outlined previously (Figures 27 and 28). It was forecast that this rate would increase. By December 1992, the network had emerged as one of the more significant technological support areas within County X. By selecting an "open system" software package for financial information, the education wide area network (WAN) began to change to different protocols. Originally a Macintosh based LocalTalk protocol was used for communication with the host student information system. This protocol is the networking support facility that is factory installed in Macintosh computers. With the connection of these networks to an Ethernet backbone, many of the networks were upgraded to EtherTalk (a higher speed protocol). With the move towards open systems and the selection of a financial package based on this environment, another protocol was required. The school district applied for and received a license for a unique world wide address protocol called Transmission Control Protocol/Internet Protocol--TCP/IP. The TCP/IP license allows the school district to access other networks, such as those at the universities and research groups. The move to these multiple protocols took place in a single year.

The school district acquired the first "64-bit" (2^{64}) computer received by a customer in western Canada (October 1992), continuing its trend of acquiring hardware shortly after it became widely available. This computer was attached onto the network. By consolidating financial reporting, as well as payroll, onto a computer attached to the network, security issues and performance of the network became very important issues to the users and the district. The strategic importance of the network and the ability to analyze, optimize, and keep the data secure have emerged as some of the highest priorities within the district. The importance of networks is projected to continue into the future as the price/performance of individual computers continues to improve. The acquisition of capable personnel who are able to manage the WAN has increased in importance in the years since the study.

Personnel Issues

The study examined the role of John as boundary spanner and Lloyd as project manager (and eventually director of computing). One area of study that does not appear to have much research in the literature is the attributes and personalities of good programmer/analysts. It has been established that computing is changing at ever increasing rates. Programmers must be able to adapt and change on an on-going basis. Studies pinpointing what characteristics are important in the selection of programmers as well as the development of interview tools for personnel selection would be a beneficial area of study and of use to school districts.

An area for further study would be the impact of technical competence, such as that provided by Lloyd. Both Boris and Rick believed that a technically competent person was not required by the manager of a computer operation. They believed that good management skills were what was required. I now believe that this position could lead to good *management* but not necessarily to good *leadership*. Unless a manager is current in the many emerging technologies, it is unlikely that the nuances of various strategies will be apparent until there is wide acceptance in the market place. This would likely result in *reactive* planning instead of *proactive* planning of networks and computing platforms.

School Scheduling

Lyle wrote a new scheduler for the SIS system. This portion of the project was written in C and, as noted by the users, the speed of the system made it possible for many iterations of various timetables to be tried by the schools. An area of research for educational administration could be the area of construction of timetables. Timetables determine which courses are available for the students to take, which teachers teach the various subjects, monitoring of class sizes, and so on. There is not a great deal of research into the rules and approaches for timetable construction within schools. Many software vendors that market student packages offer courses in timetable preparation but focus on their particular student scheduling package as compared to the general problem of timetable construction at the various levels of schools. The various techniques for timetable construction need to be identified and quantified as well as the change procedures required for staff, student, and parental involvement.

Another area related to the identification of techniques for timetable construction would be the use of rule based languages (e.g., LISP) instead of a procedural language like C. As financial restraint continues to have an impact on school systems, the ability to build student timetables with different philosophies could become a more important area for educational administrators.

Reflections

There are many events and issues upon which to reflect in this case study. These include items which would be examined in more detail if I could conduct the study again, events that become more apparent with the passage of time, and areas of further research that could be explored. These following are offered not as definitive statements but as reflections:

1. The municipal management style and organizational structure is fundamentally different than the style in education. The municipal style more closely resembles traditional bureaucracy with clear lines of authority and areas of responsibility. Education tends to reach consensus and has a more collegial approach to decision making.

2. "SIS--How much is enough?" This question was often asked by senior administrators when examining the continued support required for a software package. It is my reflection that all software requires support--even spreadsheet programs require upgrades, continual training, backup and restoration of files, and so on. The general observation is that all software requires support, internally developed systems are no different. Education continues to be a changing environment. The move to integration of special needs children into classrooms, the move to individualized learning plans, non-graded elementary schools, etc., are examples of the changing educational environment. Student information software will need to be modified continually to reflect these changes. It is my speculation that the "How much is enough?" question is a combination of the legacy of senior municipal managers that believe applications can be "installed and left running" with the desire to reduce time and effort in maintaining application software.

3. "Golden Handcuffs". The term "golden handcuffs" refers to software packages that are highly useful (the 'golden' part) but are highly specific to the hardware on which they run (the 'handcuff' part). An example of this would be a highly useful software package that runs only on a Macintosh, or only on a PC, or only on an Apple II. The extremes of completely open systems software to highly useful but highly proprietary software will continue to be a concern to those acquiring and maintaining computing environments. In general terms, "golden handcuffs" should be avoided since the future migration of these applications to other emerging hardware could prove to be very time consuming and expensive.

4. Impact Analysis. The new SIS system (scheduling, marks reporting, attendance, multi-school registrations, etc.) had a significant impact on how schools registered students, what course combinations could be offered, and so on. An area of further research could be an impact analysis on which administrative processes changed as a result of implementing the SIS system.

5. Programmers. Throughout this study I found myself reflecting about questions such as "What makes a good programmer?", "What could I look for if I were hiring another programmer?", and so on. Much of the literature I examined got into personality analysis and, as I reflect on individuals within the computing group, realized how important the type of personality is to matching people with tasks. I would investigate this area more thoroughly. I would also examine match personality traits with the Hersey and Blanchard (1988, Figure 6) model of readiness to task.

6. Technical leadership. There are those who believe that "a manager of a computing environment does not have to be technically competent but needs to be a good manager." In general, I reject this belief in an educational setting. There are simply too many fundamental changes occurring in technology which have a major impact on how computing is being conducted. If a school district is satisfied with *management* of technology instead of *leadership* in the use of technology, then there is room for this belief. Education, schools, and administration will continue to evolve during the 1990s and I believe that technically competent leadership is essential in such an environment. As illustrated by the section on "certificated vs. classified", the leader must be certified in order to participate in the Mintzberg "professional" component of the organization.

7. Systematic "Snapshots". A "snapshot" is a picture of something at a given moment in time. In order to measure change in an organization, there have to be pictures at different moments in time that can be compared. The evolution of computer networks from 1987 to the present were examined during the course of this thesis. I was in contact with other school jurisdictions which do not have "snapshots" of their networks over the years or of projected changes that are being planned. The Petruk (1986) study on microcomputers in Alberta was one of the few "snapshots" available that documented computing at a moment in time. There is a need to more systematically document the changes which are taking place in computing, both microcomputer based and mainframe based, throughout the province. This would allow the analysis of trends that are occurring. This could be an area of research.

8. "One page" Summary. During my residency, some of my colleagues had difficulty with the fairly specialized vocabulary that I had acquired over the years. I would counter that many disciplines have a unique vocabulary. At that point I would often hear comments like, "I don't want to learn all that computer stuff. When you are done, can you summarize in 'one page' what your findings were?" I will now attempt to offer a 'one page' summary. My advice to districts developing computer applications are practical suggestions to minimize the risks inherent in development:

8.1 Invest in people. No company will "look after" the district and will be committed to its long term viability to the same level as is the district. Acquire highly competent technical people who also possess a broad vision of computing and education for the future. Protect your budget in order to continually upgrade the training and professional development of your technical staff and preserve funds for prototyping new hardware and software. Do not work your key technical personnel "to death". "Good people, of course, make for good organizations. They also design good systems, at least systems that are good for them. But remove the good people and the systems collapse. Innovations, it turned out, could not be institutionalized" (Mintzberg, 1989, p. 350).

8.2 Allow "some rope". Organic decision making and dynamic structures are more academically acceptable terms, but this advice requires management to allow some latitude for computing groups to try out new ideas without being over regulated with bureaucratic controls. At the same time, accountability must not be neglected, but expect that some "balloons" may not "bear fruit" or may not do so for a number of years.

8.3 Move to "open systems" and "machine independence". This has been covered in other areas of this thesis.

8.4 Fund for the "long term". Annual contributions to a computing reserve will ensure that sufficient funds are available when needed. The more capital assets that are acquired, the larger the maintenance and capital replacement costs will be in the future.

8.5 Change depreciation expectations. Previously a five to seven year depreciation period for capital equipment was sufficient. With the rate of change in networks and telecommunications equipment, a three year depreciation schedule may be more appropriate.

8.6 Design the network. The "electronic highway" of the WANs and LANs has evolved to become the strategic infrastructure that allows for interpersonal communications in County X. Multiple protocols, security, multi-vendor equipment, and larger communications bandwidths will continue to place burdens on the network. I recommend that districts invest in both people and equipment to develop and maintain this strategic function of their organization.

8.7 Begin to share. Districts, particularly smaller districts, simply cannot afford to develop software applications or support complex computer networks. Formal and informal arrangements will be required to minimize the long term costs of computing by all districts. For example, perhaps an SIS could be made available to other districts for a nominal fee to ensure that development continues to take place, but that the total expenditure levels, at both the provincial and local levels, across the province for such applications would be substantially lower than is currently true. I continue to be very amazed at how little school districts share their computer development.

The above points are my reflections looking back on the project. My first point, invest in people, underscores how important I believe that people are to the development of computerized systems. The events of the study illustrate the crucial role that people play in computer development. The "lack of caring" exhibited by some of the groups and individuals in the study was clearly evident. I choose to summarize the importance of people with: "Love your neighbor as yourself" (Matthew 22:39).

References

Alberta Education (1992). Education in Alberta: Basic Statistics. Edmonton: Government of Alberta.

Balistreri, J. (1987). What we know about change. The technology teacher, 46(5), 3-5.

Bank, A. & Williams, R. C. (Eds) (1987). Information systems and school improvement: Inventing the future. New York: Teachers College Press.

Bluhm, H. P. (1987). Administrative uses of computers in the schools. Englewood Cliffs, N.J.: Prentice-Hall.

Bolman, L. G. & Deal, T. E. (1988). Modern approaches to understanding and managing organizations. San Francisco: Jossey-Bass.

Brand, S. (1988). The media lab: Investing in the future at M.I.T. New York: Penguin.

Bruchal, T. (1990). Innovations: strategies for implementation. Unpublished masters thesis, University of Alberta, Edmonton.

Buzzacott, D. G. (1989). An evaluation of the administrative computing structures and practices supporting school office computer use. Unpublished master's thesis, University of Alberta, Edmonton.

Carlsen, R. D. & Lewis, J. A. (1979). The systems analysis workbook: a complete guide to project implementation and control. Englewood Cliffs: Prentice Hall.

Carr, W., & Kemmis, S. (1986). Becoming critical: Education, knowledge and action research. London: Falmer Press.

County X, (1988). Junior High Boundary Study. Unpublished report.

County X, (1989). Reallocation of Information Services Mandate. Unpublished confidential report.

DeMarco, T. & Lister, T. (1987). Peopleware: productive projects and teams. New York: Dorset House.

Dunn, W. N. (1981). Public policy analysis: an introduction. Englewood Cliff: Prentice-Hall.

Edwards, C. L. (1987). The process of educational change: a staff development initiative in two school districts. Unpublished doctoral dissertation, University of British Columbia, Vancouver.

Faggin, F. (1992). The birth of the microprocessor. Byte, March 1992, 145-150.

Freedman, A. (1993). Electronic computer glossary . The Computer Language Co. Inc..

Fullan, M. (1982). The meaning of educational change. Toronto: O.I.S.E.

Gustafson, T. J. (1985). Microcomputers and educational administration. Englewood Cliffs: Prentice-Hall.

Hanson, E.M. (1979). Educational administration and organizational behavior. Boston: Allyn and Bacon.

Havelock, R. G. (1973). The change agent's guide to innovation in education. Englewood Cliffs: Educational Technology Publications.

Herriott, R. E. & Gross, N. (1979). The dynamics of planned educational change: case studies and analyses. Berkeley: McCutchan.

Hersey, P., & Blanchard, K. (1988). Management of organizational behavior: Utilizing human resources. Toronto: Prentice-Hall.

The Holy Bible: New international version (1984). Grand Rapids: Zondervan.

Johns, G. (1988). Organizational behavior: Understanding life at work. Glenview, IL: Scott, Foresman.

Kearsley, G. (1990). Computers for educational administrators: leadership in the information age. Norwood, N.J.: Ablex.

Kiesler, S. B., & Sproull, L. S. (1987). Computing and change on campus. Cambridge: Cambridge University Press.

Kosiur, D. (1992). MacWorld networking handbook. San Mateo: IDG Books.

Levin, M. (1982). Conditions contributing to effective implementation and their limits, Research in public policy analysis and management, 1, 65-111.

Long, L. (1988). An MIS case study: Zimco Enterprises. Englewood Cliffs: Prentice-Hall.

March, J. G. (1987). Old Colleges, New Technology. In S. B. Kiesler, L. S. Sproull (Eds.), Computing and change on campus (pp. 16-27). New York, NY: Cambridge.

Martin, J. & McClure, C. (1983) Software maintenance: The problem and its solutions. Englewood cliffs: Prentice Hall.

Martin, J. & McClure, C. (1985) Diagramming techniques for analysts and programmers. Englewood cliffs: Prentice Hall.

Martin, J. (1977). Computer data-base organization. Englewood Cliffs, NJ: Prentice-Hall.

Martin, J. (1984). An information systems manifesto. Englewood Cliffs, NJ: Prentice-Hall.

Martin, J. (1989). Information engineering Book I: Introduction. Englewood Cliffs, NJ: Prentice-Hall.

Martin, J. (1990). Information engineering Book III: Design and construction. Englewood Cliffs, NJ: Prentice-Hall.

Martin, J. (1990). Telecommunications and the computer. Englewood Cliffs, NJ: Prentice-Hall.

Merriam, S. M. (1988). Case study research in education: A qualitative approach. San Francisco: Jossey Bass.

Mintzberg, H. (1979). The structuring of organizations. Englewood Cliffs: Prentice-Hall.

Mintzberg, H. (1989). Mintzberg on management: inside our strange world of organizations. New York: Collier Macmillan.

Moffatt, J. (1989, October). [Interview with J. Moffatt, Employee Assistance Supervisor]. Strathcona County Schools. Sherwood Park: Unpublished.

Montgomerie, T. C. (1981). Instructional department computer support systems: A strategy and an implementation. Unpublished doctoral dissertation, University of Alberta, Edmonton.

Morgan, G. (1989). Images of organization. London: Sage.

Morgan, G. (September 1987). Gareth Morgan's images. The Financial Post.

Morrish, I. (1976). Aspects of educational change. London: George Allen and Unwin.

Newton, H. (1993). Telecom dictionary [Machine-readable data]. New York: Ziff Communications (Producer).

Petruk, M. (1986). Microcomputers in Alberta Schools -- 1986. Edmonton, AB: Alberta Education

Pugh, D. S., Hickson , D. J., & Hinings, C. R. (Eds) (1983). Writers on organizations. Markham, ON: Penguin.

Rosenblum, S. and Louis, K. (1981). Stability and change. New York: Plenum Press.

Schienbein, R. L., (1990, January). Information 2000: Computing in the future in Edmonton Catholic schools. Report presented at the meeting of senior administration for Edmonton Catholic Schools, Edmonton, AB.

Scully, J. (1989). Odyssey: Pepsi to Apple...a journey of adventure, ideas and the future. Toronto: Fitzhenry & Whiteside.

Tushman, M. L. (1977). Special boundary roles in the innovation process. Administrative Science Quarterly. 22, 587-605.

Appendix A: Glossary of Terms

AppleTalk: "Apple Computers networking software and protocols that provide the capabilities for communications and resource sharing among the computers, printers, and other peripherals attached to the network" (Kosiur, 1992, p. 506).

assembly language: "Programming language that is one step away from machine language. Each assembly language statement is translated into one machine instruction by the assembler. Programmers must be well versed in the computer's architecture, and, undocumented assembly language programs are difficult to maintain. It is hardware dependent; there is a different assembly language for each CPU series" (Freedman, 1993).

baud: "A unit of transmission speed. The speed in baud represents the number of discrete signaling elements transmitted in one second" (Newton, 1991).

bit: "(BInary digiT) Single digit in a binary number (0 or 1). Within the computer, a bit is physically a transistor or capacitor in a memory cell, a magnetic spot on disk or tape or a high or low voltage pulsing through a circuit. A bit is like a light bulb: on or off" (Freedman, 1993).

bit specifications: "(1) Size of the computer's internal word, or registers, which is the amount of data the CPU can compute at the same time. If the clock rates (16MHz, 20MHz, etc.) and basic architecture are equal, a 32-bit computer works twice as fast internally as a 16-bit computer" (Freedman, 1993).

bps: Bits Per Second. A measure of the speed of data communications (Newton, 1991).

C: "High-level programming language developed at Bell Labs that is able to manipulate the computer at a low level like assembly language. During the last half of the 1980s, C has become the language of choice for developing commercial software. C can be compiled into machine languages for almost all computers. For example, UNIX is written in C and runs in a wide variety of micros, minis and mainframes" (Freedman, 1993).

Clock speed: "Each CPU contains a special clock circuitry which is connected to a quartz crystal (same as the one in your watch). The quartz crystal's vibrations, which are very fast, coordinate the CPU's operation, keeping everything in step. CPU clock speeds are measured in megahertz, or MHz, which stands for 'million cycles per second'" (Newton, 1993).

COBOL: "(COmmon Business Oriented Language) High-level business programming language that has been the primary business application language on mainframes and minis. It is a compiled language and was one of the first high-level languages developed. Formally adopted in 1960, it stemmed from a language called Flowmatic in the mid 1950s" (Freedman, 1993).

CPU: Central Processing Unit. "The part of a computer which performs the logic, computational and decision-making functions. It interprets and executes instructions as it receives them. Personal computers have one CPU, typically a single chip. It is the so-called "computer on a chip." That chip identifies them as an 8-bit, 16-bit or 32-bit machine" (Newton, 1993).

data base management system: "The collection of software required for using a data base, and presenting multiple different views of the data to the users and programmers. The system software manages the data base, provides for logical and physical data independence, controls redundancy, and enforces integrity constraints, privacy, and security" (Martin, 1990, p. 585).

EtherTalk: A name given to AppleTalk protocols transmitted over Ethernet Media.

Ethernet: A data link protocol jointly developed by Intel, Xerox, and DEC and subsequently adopted by the IEEE as a standard" (Kosiur, 1992, p. 511).

k: Thousand (1024)

LAN: Local Area Network. "Communications network that serves users within a confined geographical area. It is made up of servers, workstations, a network operating system and a communications link" (Freedman, 1993).

LocalTalk: "The name for Apple Computer's low-cost connectivity products consisting of cables, cabling equipment for connecting computers and other devices. LocalTalk was formerly called the AppleTalk Personal Netowrk Cabling System" (Kosiur, 1992, p. 514).

Macintosh: "Series of 32-bit personal computers from Apple introduced in 1984. It uses the Motorola 68000 CPU family and an operating system that simulates a user's desktop on screen" (Freedman, 1993).

Modem: "A device that takes digital data from a computer and encodes it in analog form (modulation) for transmission over a phone line. It also performs the opposite process for incoming signals (demodulation). The term is derived from the terms *mo*dulator/*dem*odulator" (Kosiur, 1992).

MHz: Mega Hertz (million cycles per second).

open system: "Vendor-independent system that is designed to interconnect with a variety of products. It implies that standards are determined from a consensus of interested parties rather than one or two vendors. Contrast with closed system. "Open systems" often refers to UNIX-based computer systems, since UNIX runs on more different kinds of hardware than any other operating environment" (Freedman, 1993).

operating system "Master control program that runs the computer. It is the first program loaded when the computer is turned on, and its main part, called the kernel, resides in memory at all times. It may be developed by the vendor of the computer it's running in or by a third party. It is an important component of the computer system, because it sets the standards for the application programs that run in it. All programs must "talk to" the operating system" (Freedman, 1993).

OSF: "(Open Software Foundation) Non-profit organization dedicated to delivering an open computing environment based on standards. Formed in 1988, it solicits technologies from industry, invites member participation to set technical direction and licenses software to members" (Freedman, 1993).

PC: "(1) (Personal Computer) Machines that conform to the PC standard, originally developed by IBM and subsequently governed by Intel, Microsoft and major PC vendors collectively. The PC is the world's largest computer base; 1992 estimates are from 50 to 80 million units" (Freedman, 1993).

spreadsheet" "Software that simulates a paper spreadsheet, or worksheet, in which columns of numbers are summed for budgets and plans. It appears on screen as a matrix of rows and columns, the intersections of which are identified as cells" (Freedman, 1993).

TCP/IP: Transmission Control Protocol/Internet Protocol. Developed by the US Department of Defense network.
WAN: Wide Area Network. "A network that spans many geographically separated locations" (Kosiur, 1992, p. 523).

WAN: Wide Area Network. "Communications network that covers wide geographic areas, such as states and countries" (Freedman, 1993).

Appendix B: Status Reports & Correspondence

September 13, 1988

To: Manager, *the Computing Department*

From: *Project Manager*

RE: SIS Tactics Memo of August 31

With reference to your memo of August 31 as well as our meeting of September 12, I would like to summarize the items discussed:

Space

Delni: A Delni supports 8 terminal servers. Even if there isn't a spare connection for a terminal server, the *Sch1* relocation should not have to support 64 additional users.

Terminal Server: It seems to me that there already could be spare capacity to some extent on terminal servers. I suspect that this is the case. In addition, the programmers will be working in either location, not both, and perhaps something can be done to switch back and forth and use the existing ports. The relocation is not an issue of additional ports but rather facilitating "getting the job done". Since the Superintendent's Team gave the go ahead I have spent a great deal of my time examining space and facility requirements for this relocation.

513's: Unless someone wants to get into "equity", line installations et al are the responsibility of *the Computing Department*. This was brought up during the school installation. The fact that these funds reside in *the Computing Department* budgets, the funds ultimately gets charged to Education anyway and form part of the Education chargeback. The new fiscal year has just started and I would think that there must be funds for capacity expansion.

It seems that the bottom line was whose budget any and/or all costs should come from. Currently it is your position that they cannot come from *the Computing Department* budget as they are additional costs. My position is that they should not come out of the Columbia software budget where many other costs are coming from. The expenditures, as I see it, can come from:

- *the Computing Department* as an allowed over expenditure.
- the *microcomputer* software budget.
- the Superintendent's Contingency fund.
- repriorization of some budget funds.

If it is not acceptable between *the treasurer* and yourself that they should come from existing funds in *the Computing Department*, then this should be brought to the Superintendent's team meeting on the 26th for a decision. The procedure of scheduling AGT or ordering the terminals will proceed and the accounting side of this issue will be clarified on the 26th.

With regard to the "relocation" of the SIS rewrite team to Sch1, it is my understanding that:

- this is now acceptable to you as long as the programmers return to *the Computing Department* here one day/week.

- you agreed that the tasks that they are working on when they are in this building would not change.

- you wish to reevaluate the need to stay in *Sch1* in March. To this I expressed a concern that I did not want the Board to agree to a one year lease and the expense of relocation for Oct. - March.

It has always my position that relocating the programmers both minimize the interruptions and allows for the whole team to be in one place and therefore reduce the decision making cycle time. Part of the success of Phase I was the fact that it was possible to resolve issues quickly by having all of the participants proximal to each other. I have no vested interest in "getting the team" out of the building. Indeed, it has added another task to the things that I am doing to arrange for the many things that need to be done. My concern is to improve the success of the project wherever I can. As I stated on the 13th, the team can remain where they are and I will continue to monitor their progress as project manager. I would have appreciated having your concerns raised before presenting our report on the August 29.

Training

While I agree that the *Supervisor, Computers* did a tremendous job of training in Phase I and it would be desirable to have her again in Phase II, it was pointed out that *the Ass't Principal* is now in place for the entire year. While I am exploring what ways he can assist in the training of Phase II, training is part of the project and plans should be made to meet the timelines. The major reason that Phase I training occurred as it did is that no one else was available.

I.S. Technical Support

I agree that key individuals involved in the SIS rewrite also have major technical responsibilities. These things have been known for a number of months. I have difficulty in setting aside 10,000 - 30,000 of the *microcomputer* budget for specialized technical support. If you wish to have this presented to the Superintendent's Team, I will do so. You will recall that chargebacks for technical support were not done when Summer Games made use of these people, and was not true when extra programming support was required for the FIS rewrite. I have been consistent in my efforts to minimize the workloads of the technical staff and have invested a great deal of my own time into assisting in them where I can. I continue to support the technical side of the Transportation software.

In terms of the "technical demands growing to a point of threatening the SIS time lines", I am proceeding under the assumption that the FTE's assigned to the SIS project as presented to the August 29 meeting were correct. This means that only the unallocated portions of the technical staff not assigned to SIS is left to work on technical problems. If technical problems occur requiring extra resources, then these are outside of the SIS project.

Summary

Although a case could be made for some of the costs of the relocation be supported outside of *the Computing Department* budget, I do not support paying for the Delni's, 513's, etc. on your list out of the "*microcomputer*" budget. Whether the programmers use the line from the sub basement or Brentwood still only uses one line each. You were present at my presentation to the Board when I asked for these funds. You budgeted for things that were needed within the SIS rewrite and I budgeted and defended things required for the R & D side of the rewrite. In the Board review there was a $60,000 reduction. It was *the Computing Department's* choice that this came out of the SIS rewrite.

It is my understanding that the responsibility of the rewrite resides with *the Computing Department* and the responsibility of myself as Project Manager is to ensure that it gets done.

pc *Associate Superintendent*
Treasurer
Ass't Principal
Coordinator

SIS Reporting for the Period to 19-Oct-88

1.0 Highlights

•meeting of the "large" SIS group took place 12-Oct.

•at both the SIS meeting and Junior High principals' meeting:

-terminal response time was noted as an issue. It was felt that the length of time for the terminal to respond was too long

-secretarial time was deemed to be too low due to SIS. The elementary principals are currently circulating a questionnaire among themselves. The extra work loads in the office are clearly being attributed to SIS. The issue is high on everyone's mind.

•*Seconded administrator* has visited each senior high and most junior highs on scheduling/marks.

•*coordinator* has spent 15 days since September on non-SIS material (7 were projected). It is anticipated that the amount of time spent on non-SIS will diminish as the new technical support contractor picks up tasks. *Programmer*$_2$ has spent 8.5 days since September on non-SIS material. It is expected that this will halt.

•as training and Phase II/III issues are examined, it is apparent that plans for support for 1989/90 will soon have to be formulated, particularly for the Sept-October time window.

•*Sch1* and *Sch2* are back to 2400 baud, *Sch3* still having problems. The modem has been replaced and problem escalated to Edmonton Tel.

2.0 Decisions

2.1 Training

Training material attached. Recommend a Certificated person for Feb. 1989 to June 1990 (August is preferable but not clear as to the status of how this could be done) to coordinate training.

**SIS Reporting
for the Period
to 9-Nov-88**

1.0 Highlights

- move to *School* complete. All terminals and printers working.

- support for Phase I at 35 days of 40 allocated. This appears to be slowing.

- modem at Sch2 continuing to be monitored. "One good day". *the Computing Department's* has escalated this with Edmonton Tel. and *Programmer3* is monitoring.

SIS Reporting
for the Period
to 30-Nov-88

1.0 Highlights

• very positive SIS meeting/demo of the course/section software. Excellent progress being made.

• 78.5 days is the highest overrun and indicates of a trend.

In the past this was offset by programmers working overtime and the contract programmer working at the front end of the project. It is not realistic to expect the programmers can continue to work the same amount of overtime since they are becoming "run down". The rate of being "siphoned off" to other projects and tasks instead of SIS has continued.

It is felt by *Computing Department* that Part A is still within the capability of meeting the target deadlines. Since the tasks are so large, it is difficult to determine the nearness of completion until the task is nearly complete. This means that by about mid-January the timeline estimates will be accurate and it will be known if Part A is in difficulty. Erosion to other project tasks must cease.

•Task 207 (course requests) is overrun by 31 days.

The original estimate was, in all likelihood, too low.

Using up much of the equalization factor and administration factor.

Difficulties have been in the course master file update and maintenance.

•new Alberta Education regulations will likely cause additional fields to be incorporated into both the existing Phase I as well as Phase II (language of instruction, etc.). These have not been incorporated into the timelines but are brought forward for information.

•a plan for school opening in September 1989 needs to be devised prior to the budget cycle.

SIS Reporting for the Period to 25-Jan-89

1.0 Highlights

•Task 207 (course request/maint) generally done!!

•Junior High (*Sch1*) involvement Dec. 20,21. Resulted in course/section homeroom update added to the overrun but ensured ease of use for the Junior Highs.

•Senior Highs and *Sch1* all represented for .5 day inservice (Jan 16) on course request/maint.

•good news is that overrun back down to 106.3 days (Task 110 done - 4 days, 202 done, overrun now included in time totals).

•since Dec 15, 6.0 days for *Programmer1* to Payroll, ~22 days for *Coordinator* for split and budget.

•course request/maint. documentation about 80% complete (*Ass't Principal*).

•February 1 slated for "startup" for 1989/90 school year for registrations.

•*consultant's* study into Senior High offices underway.

•4 days holidays *Programmer1*, 6 for next reporting period.

•*Consultant* -7 days (allocated to Report Writer interface). Task 211 is likely low in estimate and may be approximately 40 days. Much of the time likely to be made up prior to March.

•potential "hit list" of deferrable tasks identified: 35 days for Phase I in Part A, 43 days for Part A can go to Part B and other people. *Summer Student* will add about 70 days beginning May 1 (training, documentation).

2.0 Concerns/Issues

•level of Administrative involvement at Sch4 (Ass't Principal working on).

•new report card needs involvement from Instructional Services.

•need to minimize *Coordinator's* involvement in non Scheduler related tasks through to March.

•need to examine support work to be performed for Human Resources by *Programmer1*.

•Phase III course history and prerequisite checking needs to be a high priority for 89/90.

•Summary of Enrolment may need major editing depending upon how the schools dealt with transfers. This will have to be complete before June.

•demo for Supt's Team?

"Doability Probability":

(Part A minus the scheduler Mar 1) *Coordinator* 100%, *Project Manager* 95%.

All tasks on project for Mar. 1 - very low.

Schools can start course entry and startup, scheduler can be completed after Mar. 1.

TECHNICAL EVALUATION OF THE *MICRO* SCHEDULER
(For Use With The SIS System)

Technically the Scheduler from *Micro1* Computing Services is a good product. It behaved well during evaluation. In our test runs it even out-performed the *EXISTING SYSTEM* scheduler by approximately 5-10% in the number of successfully scheduled students. This appears to be mainly due to the fact that *Micro1*'s scheduler will re-evaluate previously successfully scheduled students - where as *EXISTING SYSTEM* was designed to be a one pass scheduler.

However we discovered a number of deficiencies with the *Micro1* system that would prevent a seamless integration with the existing SIS system. Only the technical difficulties and problems we encountered will be highlighted. The functional difficulties are to be reported by *John*.

•Section numbers are resticted to a 2 digit number where SIS keeps a 4 character string. A special file would have to be set up for each school to allow them to convert the sections number when data is moved back and forth between *Micro1* and the VAX. However because our schools use the section number to keep other information this information would still be lost when the data is transfered to *Micro1*.

•the information kept by SIS did not map cleanly to information required by *Micro1*. Considerable manipulation needed to be done before the data could be downloaded to *Micro1*.

•because of the data differences between the two systems it would not be feasible to have the schools add or modify student, course, section or school information in *Micro1* and upload it later to the VAX. Not only would information required by SIS be missing by the information collected by *Micro1* would have not undergone the stringent checking required by SIS. The only realistic way to overcome this problem would be to have all new or changed information be added in SIS and then download it to *Micro1*. The time and extra work this would take could seriously impede the scheduling process.

•transfer time to get information between SIS and *Micro1* was approximately 4 hours the first time for 1400 students, then subsequently 30 minutes (1-2 students). This may not be acceptable to the schools.

•individual simulations took 8-10 minutes (1200 students) where as on *EXISTING SYSTEM* took 3-5 seconds. The problem appears to be that the scheduler loads all course/section information and must look at all the student in the school to find the one that needs scheduling. *EXISTING SYSTEM*, on the other hand, only loads the required course/section information and is able to zero in on a single student directly. If you add to this time to the time it takes to transfer student information from *EXISTING SYSTEM* to *Micro1* it would force the schools to change they currently handle in-person registration.

•the *Micro1* scheduler does not allow the user to bypass scheduling changes. Once you do a simulation it is permanently written to the student file. *EXISTING SYSTEM*, on the other hand, lets you "play" with different combinations to get the best fit for the student and it does this

quickly (see above). But if you happen to really mess things up you always have the option to restore the original schedule and try again. This is a nice safety feature.