Chapter 22: Practical Case Study

This case study is based on an actual IT program. The consultant was contracted to develop the management information system (MIS) program methodology and reporting system. Despite having established new IT program management guidelines, this program followed the old methodologies and employed the traditional mindset encountered in typical IT programs.

Company Background

This national corporation had been purchased twice within a 3–year period, with all employees and assets included in the transaction. During the second takeover, many of the IT staff members took positions with other firms in the area. The parent corporate MIS department hired a new Chief Information Officer (CIO) to head the MIS organization. He in turn brought in a new senior manager to assist in overseeing the MIS staff. Simultaneously, the corporate MIS department hired someone from outside to be the CIO, but he was relegated to the position of vice president (VP) of MIS at the division level.

Corporate MIS Structure

The corporate structure consisted of the parent corporation divided into several divisions. A chief executive officer (CEO) and his staff headed each division. On each staff was a CIO who ran the MIS department. At the parent level was an MIS department that set the overall strategy for the divi

sions. The parent MIS department provided service professionals who were responsible for help desk support, equipment configurations and installation, maintenance, systems support, network support, creation of new user accounts, monitoring of software usage, and maintenance of the MIS inventory. These professionals were assigned to a division full time but were paid and evaluated by the parent MIS department.

At the division level, the CIO reported to the CEO and separated his organization into two major departments:

1. The Software Development and Maintenance department was responsible for new application software development and the maintenance and support of existing software applications, training, and customer services (mini help desk). The new senior manager of MIS, brought in by the CIO, headed this organization.

2. The Business Development and Technical Services department consisted of a consulting organization, technical support and development, and business analysis. The VP of MIS was brought in by the corporate managers who headed this department.

Conflict

The CIO and his manager had no experience in this industry, whereas the VP of MIS came from a similar company and had a vast amount of experience. Soon two distinct factions evolved within the MIS staff, and the board of directors (BOD) became heavily entrenched in trying to sort out which faction was pointing them in the right direction (Figure 22–1). The CIO did not want his authority compromised and continued to ensure the board that all was well and on schedule.
The user hardware consisted of personal computers (PCs) running on Novell local area networks (LANs). PCs acting as router hubs connected each LAN to the central database. The corporate service organization mandated that all users would use a Compaq 486 PC with Intel 33 MHz chips. The router hubs were Compaq 386 PCs with Intel 33 MHz chips. The database consisted of approximately 30 gigabytes of storage on a disk farm.

**Software Environment**

Most of the existing applications were developed in Clipper, an outdated suite of development software. New applications were developed in Sybase and Powerbuilder. No documentation of the Clipper environment existed, and seasoned professionals, mostly contractors, were assigned to support that application set. Tenured staff members were used to supervise the remaining contractors doing development in Sybase and Powerbuilder.

**Corporate Service Professionals**

Corporate service professionals provided a variety of services to the division. During one period of growth, while hiring some new contractors, a need arose for additional PCs. The division argued that the services group should order new 66 MHz models to improve productivity in the MIS area. When the new PCs arrived, the application development area improved, but the greatest influence was how fast the user applications ran on the new hardware. The users were brought in to see the improvements in response time and readily agreed to purchase the newer models.
However, when MIS went to the service group to place the order for newer models, they were told that inventory control practices for maintenance and spare parts prohibited purchase of newer models until the corporate approval was given.

As new contractors and staff members came on board, they would set up their new accounts by copying applications from the application database server. The support group would monitor the number of licenses being used and, if the quota were exceeded, would deny access to the software. The contractors were expensive to employ and were kept waiting until budgets, approvals, and purchasing and license agreements were settled. This often took several days to resolve.

**Role of Business Services and Technical Support**

The technical support organization was evaluating operating systems for future MIS applications in the division. Windows NT and Windows 95 were among the candidates. However, they were primarily interested in the technical aspects and seemed unconcerned with the ramifications of software availability for these new operating systems. No transition plan existed to migrate existing applications, nor was there availability of development software on the new platform.

In the meantime, the consultants and business analysts were interfacing with the user community, defining new applications to be built by the software development group. They busily established project schedules and prioritized the development efforts. When the software development group hesitated and pushed back the schedule because of personnel shortages, chaos erupted in the user community.

**Role of Software Development and Maintenance**

This organization was under heavy user pressure. Peak periods were during the day when the users were constrained to a specific amount of time to process their work. Failure to do so resulted in extreme customer pressure and missed deadlines. The equipment in the Novell network was so outdated that the users began experiencing severe gridlock in the network. New applications that were recently installed had to be halted, and the users had to go back to manual or old methods of performing their work.

More chaos erupted, and everyone was fighting fires. Experiments were made with the router hubs; the changes decreased performance and made the MIS department look as if they did not know what they were doing. They reverted back to the old setup and promised to correct the situation. To complicate matters, members of the MIS staff continued to break ranks, and soon the contractors outnumbered the permanent staff. The lack of tenured staff members had everyone fighting fires, rescheduling program and project implementations, and working overtime. During this period of conflict, the CIO committed to the BOD that a new program, urgently required by one of the user groups, would be implemented on schedule and within budget.

**Program**

This program was initially set up using the new program management guidelines and predicted an implementation schedule that would last 6 months. The integrated program schedule was documented using Microsoft Project, in which fully loaded costs were used to determine program costs. Figure 22–2 shows the program schedule, and Figure 22–3 outlines the program costs.
The program schedule was reviewed with the CIO and his staff before presentation to the BOD. Cost and schedule were reviewed, and it was determined that the cost savings would net a return on investment (ROI) of approximately 35%. The program would pay for itself in less than 3 years. The total cost estimate was approximately $275,000. The program team consisted of a program manager, team leader, consultant, business analyst, systems analyst, contract programmers, MIS operations, and trainers. Not all members of the team were assigned full–time responsibilities.

Program Execution

The program started off well. Gathering data for the functional and technical requirements went smoothly, and the user group was satisfied. The program team went right to work on the screen design for the user interface and proceeded to change record structures to accommodate the new data fields. The analysts and programmers were delighted with the new functionality that they could provide to the users and eagerly showed everyone in
the MIS department their progress.

The user group would have a vast array of options from which to choose, and this would enable them to provide better service to the customer base. Trainers joined the team and proceeded with the same vigor as the analysts and programmers. They developed an impressive training package that showed that they were proud of their accomplishments. During this transition, the CIO was reporting to the BOD that all was well and on schedule. The BOD members representing the user groups took it for granted that the MIS was keeping their management abreast of the program content as it progressed. However, they did not realize that the user group had little free time to spend with their MIS counterparts. The program team thought that this was business as usual and usually did not consult with the user group on screen layouts or option functionality.

Training was scheduled and started without any member of the user group having seen the new screen designs or the supposed improvements in functionality. When the actual training sessions began, the users were disappointed with the software. This vast array of options required them to plod through several screens that were too time consuming. The user group wanted to speed up the process so that more work could be done in shorter period of time and consequently gain improvements in customer satisfaction.

Results

Many problems faced the MIS organization. Some had to do with hardware and personnel resources. Other problems arose out of politics and job security. The program was never completed, and the software was not installed. The users flatly refused to accept it. The functionality was not what the users wanted, and a software rewrite would take too long. More important things were now the focus. The capital influence on ROI was about $375,000. More importantly, the user community experienced staff increases and constant pressure as the customer base increased along with the proportional workload.

User confidence in the MIS department had eroded, but a significant backlog of work still had to be accomplished. Somehow, the MIS management structure remained in place, and they continued to make promises they could not keep. The management and their staff continued to spend significant amounts of money on contractors, consultants, and enabling technology that did not blend into a coherent solution for the end-users.

Lessons Learned

Considering the documented history of this program, it was doomed from its inception. Too much time was spent on politics and keeping management in the dark concerning the true nature of the problems facing the MIS department. The company became caught up in the everyday process of putting out the next fire. Someone, somehow, should have evaluated the risks associated with this program and other programs that were under development at the time. Assessment of the companys position relative to the coordination of resources, technical direction, staffing shortfalls, and lack of user involvement would have alerted someone that the project was not going well.

Risk management is a difficult process. It is not a process of excuses but of identifying the obstacles to successful completion of an assigned task. The difficulty is in the assessment of its influence in terms of cost, schedule, and probability. The more difficult task is the realization that a good program manager should have alternate solutions to abate the risk and identify how the cost, schedule, and probable influence on a program schedule could be reduced.
Risk Analysis

Risk can be associated with a root cause. A risk abatement strategy can be formulated by bypassing the symptoms and focusing on the root cause. Associated risks can be minimized through identification of several risks that have the same root cause and elimination of the root cause.

In the program discussed in this case study, the same root cause (changing requirements) was documented for seven of the risks identified. The potential cost damage of these risks was in excess of $100,000. Elimination of the root cause by establishment of the baseline configuration would diminish the probability of such an extravagant cost overrun. Several instances existed in this program of multiple risks associated with one root cause. A few fixes could have solved many problems and may have led to successful completion of a difficult program in adverse circumstances. For example, the lack of human resources was cited three times and carried a potential cost of $180,000. Mitigation of this risk by the proper allocation of personnel could have saved money and increased the projects chance for success.

The program should have been on hold until these problems were resolved. The MIS department could have saved its reputation, the user community would have been alerted to its own internal requirements, and the corporation would not have made a bad investment of time, resources, and capital.

Using RiskTrak, the consultant spent a couple of hours identifying the risks associated with this program and honestly evaluating the influence in terms of cost and schedule. The surprise came when the reports were generated and the indication of potential cost and schedule influence was realized. After careful consideration, the consultant realized that using a risk management tool could have saved the corporation several hundred thousand dollars during his 6−month tenure across a variety of programs. The cost and schedule reports indicated that not all of the potential damages to cost and schedule would have occurred. The risk management software factors in the percentage of probability and damage to forecast predictable results if the risks identified are not managed. The risk−identification process triggered and freed the program managers mind from only considering the risks associated with the most recent program.

If this large corporation had used the RiskTrak preproject analysis before making a decision to proceed with this project, they would not have launched the program. For a potential ROI of less than $100,000 per year, the risk in terms of cost, time, and schedule clearly outweighed the gain, especially when considering a possible schedule slip of approximately 18 months and a cost overrun in excess of $375,000 on a 6−month, $275,000 project. It would have taken over 2 years to develop this project and over 6 years to retire the investment, which would not make it a cost−effective project. If the RiskTrak risk engineering had been applied and only the top two risks (e.g., changing requirements and lack of human resources) were identified and effectively managed, a potential slip of 260 days could have been avoided and over $180,000 could have been saved. Although this would still leave a substantial cost and schedule overrun, the potential for additional savings existed in which further risk management could have been applied to mitigate the potential losses in time and money. RiskTrak postmortem analysis shows that poor planning and inconsistent data were key to the failure of this project (from Services and Technology Group, www.stgrp.com).

Acronyms

**A–C**

**ACAP**  Analyst Capability

**AEXP**
Application Experience

AI
Artificial Intelligence

ANSI
American National Standards Institute

API
Application Program Interface

ATD
Actual to Date

CASE
Computer–Aided Software Engineering

CCB
Configuration Control Board

CCSOM
Computer Center Software Operational Manual

CDR
Critical Design Review

CDRL
Contract Data Requirements List

CEO
Chief Executive Officer

CFD
Control Flow Diagram

CFO
Chief Financial Officer

CI
Commercial Item

CIO
Chief Information Officer

CLIN
Contract Line Item Number

CM
Configuration Management

CMM
Capability Maturity Model

COCOMO
Constructive Cost Model

COM
Common Object Model

CORBA
Common Object Request Broker Architecture

COTS
Commercial–Off–the–Shelf

CPM
Critical Path Method

CPU
Central Processing Unit

CSC
Computer Software Component

CSCI
Computer Software Configuration Item
**CSDM**  
Computer Software Development Methodology

**CSOM**  
Computer System Operator Manual

**CSU**  
Computer Software Unit

**D**

**DADP**  
Domain Analysis and Design Process

**DBDD**  
Database Design Document

**DBMS**  
Database Management System

**DCE**  
Distributed Computing Environment

**DCI**  
Distributed Computing Infrastructure

**DCOM**  
Distributed Common Object Model

**DD**  
Data Dictionary

**DDD**  
Domain Data Dictionary

**DDL**  
Data Definition Language

**DDM**  
Domain Dynamic Model

**DE**  
Domain Engineering

**DFD**  
Data Flow Diagram

**DFM**  
Domain Functional Model

**DID**  
Data Item Description

**DII**  
Dynamic Invocation Interface

**DIM**  
Domain Information Model

**DOD**  
Department of Defense

**DOE**  
Distributed Object Environment

**DOI**  
Distributed Object Infrastructure

**DOM**  
Distributed Object Model
**DOT**
Distributed Object Technology

**DPM**
Domain Prototype Model

**DSI**
Delivered Source Instructions

**DSOM**
Distributed System Object Model

**DSSA**
Domain–Specific Software Architecture

---

**E−I**

**EAC**
Estimate At Completion

**EC**
Estimated Cost

**ECP**
Engineering Change Proposal

**EIA**
Electronic Industries Association

**ELOC**
Estimated Line of Code

**ER**
Entity Relationship

**EV**
Earned Value

**FCA**
Functional Configuration Audit

**FP**
Function Points

**FQT**
Formal Qualification Testing

**GUI**
Graphic User Interface

**HTML**
Hypertext Markup Language

**HW**
Hardware

**HWCI**
Hardware Configuration Item

**ICAM**
Integrated Computer–Aided Manufacturing

**ICT**
Intelligent CASE Tools

**IDD**
Interface Design Document

**IDEF**
Cam DEFinition
IDL  Interface Definition Language
IEC  International Electro−technical Commission
IEEE Institute of Electrical and Electronics Engineers
IIOP  Internet Inter−ORB Protocol
I/O  Input/Output
IORL  Input/Output Requirements Language
IPT  Integrated Product Team
IRS  Interface Requirements Specification
ISO  International Standards Organization
IT  Information Technology
IV&V  Independent Verification and Validation

LOC  Lines of Code
LOE  Level of Efforts
MANPRINT  Manpower and Personnel Integration
MIS  Management Information System
MM  Man Month
MMI  Man−Machine Interfaces
NATO  North Atlantic Treaty Organization
NDI  Nondevelopment Items
NDS  Nondevelopment Software
OAM  Object Analysis Model
OBM  Object Behavior Model
OCD  Operational Concept Document
ODF  Object Definition Language
OIM  Object Information Model
OLE  Object Linking and Embedding
OMA  Object Management Architecture
OMG  Object Management Group
OML  Object Management Library
OO  Object Oriented
OOD  Object-Oriented Design
OODB  Object-Oriented Database
OODBMS  Object-Oriented Database Management System
OODM  Object-Oriented Design Method
OOM  Object-Oriented Methodology
OOP  Object-Oriented Programming
OODS  Object-Oriented Structured Design
OPM  Object-Process Model
OSF  Open Software Foundation
OSI  Open System Interconnection
OTS  Off-the-Shelf

P–S

PC  Personal Computer
PCA  Physical Configuration Audit
PERT  Program Evaluation and Review Technique
PIN  Personal Identification Number
PSM  Practical Software Measurement
P−Spec
Process Specification

QA
Quality Assurance

QAS
Quality Assurance Section

RDM
Requirements Definition Model

RE
Re−software Engineering

RELY
Required Software Reliability

RFP
Request for Proposal

ROM
Read−Only Memory

RSE
Reverse Software Engineering

RSO
Reusable Software Objects

RT
Requirements Tracer

RTE
Run−Time Environment

RTL
Run−Time Library

SA
Structured Analysis

SCR
Software Cost Reduction

SD
Structured Design

SDD
Software Design Document

SDF
Software Development File

SDL
Software Development Library

SDP
Software Development Plan

SDR
System Design Review

SECP
Software Engineering Conversion Plan

SED
Software Engineering Design

SEDD
System Engineering Design Document

SEDP
Software Engineering Development Plan

SEI
SysDD
System Design Document
SysRS
System Requirement Specification

T-X

TRR
Test Readiness Review
UC
User Cases
UI
User Interface
VDD
Version Description Document
V & V
Verification and Validation
WBS
Work Breakdown Structure
WWW
World Wide Web
XML
eXtensible Markup Language