Bridging the Technology Generation Gap: Upgrading a Network Management Application to a New Technology Base

Jay S. Lark

Teknekron Communication Systems, Inc.
jlark@tcs.com

Pierre Osborne

Stentor Resource Centre, Inc.
ozzy@tcs.com

Abstract

NMCS is a network management system which has been developed to manage dedicated fractional-T1 networks, allowing end-users such as banks and government agencies the ability to dynamically manage the allocation of their dedicated communication facilities. The first version of NMCS was delivered on a PC platform, and was a successful and well-accepted product. A new version of the system was designed to address some of the shortcomings of the first version.

NMCS II is a significant upgrade to the first NMCS product. In addition to extending it functional capabilities, NMCS II is built on a different technology base from its predecessor: the hardware and operating system, application architecture, user interface, and database have all been upgraded to a new technology generation.

NMCS II is now installed and operational at several customer sites throughout Canada. While the product has passed its formal acceptance, a number of acceptability issues were present in its first release. Many of these issues derive from the changes to the technology base, which we call the "technology generation gap."

We look at several of these acceptability issues which we encountered, how these issues relate to the technology generation gap, and how we are addressing these issues going forward. We conclude with a discussion of lessons we have learned to help avoid falling into the "gap" in the future.

Introduction

Stentor, through Bell Canada and the provincial telephone companies in Canada (Telcos), offer a national fractional T-1 service known as *MEGA*Stream, targeted at companies and institutions which use telecommunications in their mission-critical applications. The *MEGA*Stream service is a combination of hardware, management systems, and customer service components which form a unified solution to end-users needs for dedicated bandwidth.

An important component of the *MEGA*Stream service is the Network Management and Control System (NMCS), a network management system which allows end-users to control the allocation and routing of their bandwidth to match their business needs. It also provides additional features such as testing of customer circuits and disaster recovery. NMCS is also used by the Telcos to configure and test the physical elements in the network, and provide additional support to the end-users through a unified view of the customer's network.

The first version of NMCS ran on a PC (referred to as NMCS-PC in the sequel). It was first introduced in 198x, and it was a successful product. However it was a victim of its own success, as it was unable to evolve fast enough to meet customer demands for increased capacity (larger networks) and services (new network equipment and services). One of the major impediments to evolution was the technology base upon which NMCS-PC was developed:

- a segmented hardware architecture, which required application code constructs to manage memory segments
- QNX operating system, which was not readily portable to other hardware platforms
- a proprietary database system which did not allow easy export of management data
- character-based user interface, which limited the amount of information which could be managed on the display

NMCS II Requirements

In 1991 Stentor Resource Centre Inc. (SRCI), the development organization for Bell Canada and the Telcos, contracted with Teknekron Communication Systems, Inc. (TCSI) to build a new version of NMCS. NMCS II was designed to address the limitations of NMCS-PC and provide significant new services and features. The requirements for NMCS II stated that the new system should be compliant with many formal and de facto standards, including:

- OSI communication interfaces
- OSI Network Management Forum object classes
- POSIX
- Sun SPARC workstations
- Ingres and SQL
- C or C++

The requirements mandated a "modular and scalable hardware/software architecture" using managed object (MO) classes and instances. It was required to support multiple, simultaneous users logged-in from geographically-remote sites. The user interface was to be graphical, and provide a distinctive look-and-feel.

The functional requirements included the definition of 40+ Managed Object classes, where each instance of a class would typically contain 20-40 attributes and define 10-20 operations. These functional requirements were a significant superset of the functionality implemented in NMCS-PC, supporting several new Managed Object classes and behaviors corresponding to new network elements.

The functional requirements were organized into four categories:

- Configuration Management:
  MOs which model the physical and logical elements of the managed network, such as multiplexers, link, and circuits, along with the operations to configure, audit, and test these objects. The components of the managed network are referred to as network elements, or NEs.
- Fault Management:
  MOs to represent the alarms and other events which are signalled by the managed network, and logging tools to maintain a record of these events and an audit trail of user actions.
- System Management:
  MOs to represent various aspects of the NMCS II system itself, such as the authorized users and data-

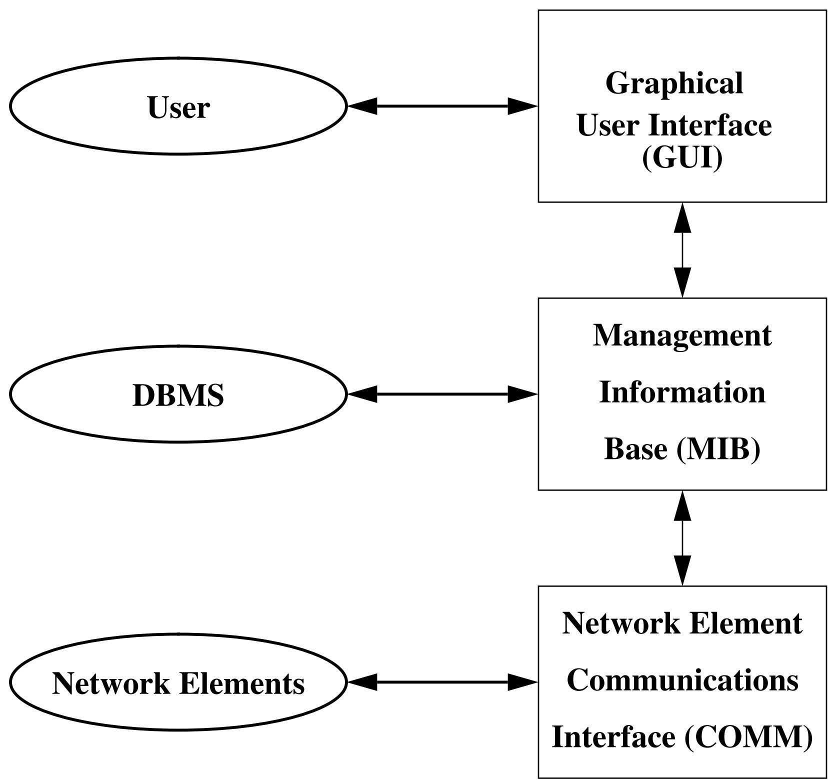
base options.

- Utilities:
  Tools such as calculator, notes, and Unix windows.

NMCS II Architecture

NMCS II is based upon a distributed process architecture, summarized in Figure 1.

Figure 1: NMCS II System Architecture



The functional components, represented by the boxes on the right, are built on top of NMS/Core™, a distributed, client-server, object oriented development platform developed and licensed by TCSI. The ovals on the left side represent the elements external to NMCS II.

Management Information Base

The heart of the NMCS II system is a Management Information Base (MIB), an object oriented repository of all Managed Object instances in a customer installation. The MIB is responsible for concurrency management (for multi-user operation), validation of all changes to MOs, and for implementing the main functions of the system: monitoring and configuration of the Network Elements (NEs). The MIB uses a relational database (Ingres) for persistent storage of its objects.

The MIB is implemented by two distinct programs. The Master MIB (MMIB) is a single process which is the “owner” of all managed objects. It is the only process which can update the database, and provides a single point to manage concurrency and consistency. The MMIB implements a “lazy” object locking mechanism, which allows an form on the GUI to be modeless, *i.e.*, a user can modify any object without requesting an object lock beforehand.

The Slave MIB (SMIB) program functions as a "write-through cache," providing high performance access to view MOs without loading the central MMIB server. The SMIB reads and caches objects from the database on start-up, and can then respond to any read request from the GUI from its local cache. Any attempt to modify an object passes through the SMIB to the MMIB, which validates the request before updating the database. Changes in the MMIB are communicated by events to all SMIBs, which update their cached data accordingly. Each NMCS II installation can support an unlimited number of SMIB processes.

The MMIB and SMIB use the same internal architecture and share much of their base code, which minimizes their development cost.

Graphical User Interface

The Graphical User Interface (GUI) communicates with the users of NMCS II through a set of forms (windows) which are displayed on the user's workstation. The GUI is an X.11 client, providing a variety of list, detail, and graphical forms to view and modify MOs. The forms are built using OSF/Motif widgets, but implement a look-and-feel quite different from that mandated by the "Motif Style Guide". The GUI uses uid files (a Motif standard representation for form/widget layout) to simplify the development process and provide for easy field upgrade of specific forms.

The GUI forwards requests to an SMIB process to fetch and modify MOs, and to invoke operations on MOs. Each NMCS II installation supports an unlimited number of simultaneous users, each communicating through their own GUI process to the shared MIB.

Network Communications Interface

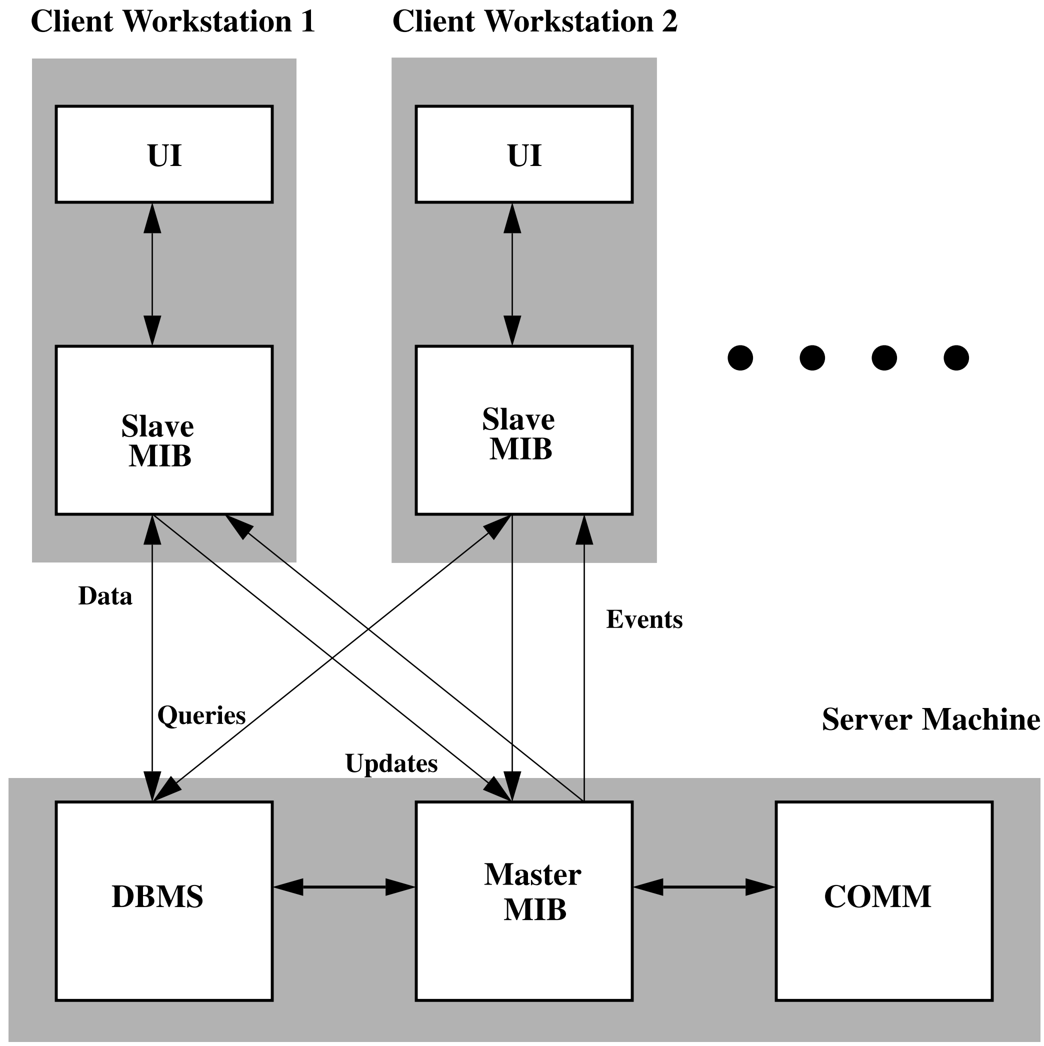
The MIB communicates to the NEs through the COMM process. COMM's main responsibility is to drive the serial line connected to the managed network, translating commands and responses between the NMCS II internal format and datagram format used by the NEs. The MIB uses a non-blocking message-based interface to communicate with the COMM, and can process other requests while waiting for a response from an NE. Each NMCS II installation has a single COMM process.

Process Structure

The MMIB, database, and COMM processes typically reside on one or more server machines located in a central office location, for ease of maintenance by telephone company (Telco) personnel. In contrast, the client workstations can be located in customer sites, and connected to the server machine by an X.25 interface. Each client workstation contains a GUI and an SMIB process, and there are no fixed limits to the number of client workstations. (See Figure 2).

All processes communicate using a single universal message format, and use XDR for encoding data. The message format is proprietary, and closely matches the format of internal data structures. Distribution and location/binding of processes is done through simple configuration files, and is quite flexible; there is no fixed relationship between processes and processors, and processes can be combined or separated freely.

Figure 2: NMCS II Process Structure



Technology Generations

To the authors' best knowledge, NMCS II is the most sophisticated and feature-laden network management systems for Fractional-T1 networks in commercial use. Delivering a system of this complexity and magnitude was a difficult undertaking, for both the development and deployment teams. The NMCS II system is based upon a major technology base upgrade from that used in NMCS-PC. That technology base includes many components, which are briefly described below.

One of the underlying difficulties we experienced with many areas of the technology base was "generation creep" — changes to the supported version of a technology which occurred during the development process.

<I am expecting additional feedback on this section>

Application Architecture

NMCS PC was designed and developed using traditional procedural programming techniques, although the designers had started to think of the application in object oriented terms. This perspective was evident in the functional requirements, which specified managed object classes (the "leaf" classes) but did not inheritance or other techniques to manage design complexity and promote re-use.

NMCS II was designed using object oriented design and analysis techniques, and is implemented on NMS/Core, a TCSI-proprietary object oriented development platform. The definition of the object model and inheritance hierarchy was an iterative process between Stentor domain experts and TCSI architects. It appears that object oriented analysis techniques are not a panacea, and object oriented designs are only as good as the skill of the practitioners.

Hardware Platform

NMCS II is delivered on Sun SPARCStation platforms, compared to an AT-class machine for NMCS-PC. This decision did not cause problems during the development process, but did have significant implications for delivery. The major issues included: cost to deploy the system, installation and maintenance of new hardware, and desktop footprint.

Operating System

NMCS-PC was implemented on QNX, a UNIX-like real-time operating system for PCs, NMCS II is built upon SunOS. The SunOS product is solid, but we experienced issues with revision upgrades, from 4.1.1 to 4.1.3. A port to Solaris 2.x is on the horizon, with unknown but potentially significant upgrade costs.

User Interface

NMCS-PC's user interface was based on character graphics. It was fast and easy to use, but was limited in the amount and complexity of information it could display and the ability to navigate between parts of the system. The NMCS II GUI overcomes these problems through the use of multiple windows, graphics, multiple display formats, and enhanced navigation capabilities which are common to many Motif applications. However, we have experienced many problems with moving to a windows-based interface, some generic and some specific to Motif.

- user discomfort with a mouse
- user difficulty managing multiple windows
- insensitive fonts, keybindings, and other "trivial" concerns
- difficulties upgrading from Motif 1.1 to 1.2

Database

NMCS II uses the Ingres relation database, compared to a proprietary, in-core database used in NMCS-PC. This decision has major advantages in the design of the database schemas, producing reports, allowing the database engine to be distributed to high-performance servers, and allowing integration with other Stentor data systems. However, we had to overcome several problems to realize these advantages:

- Ingres does not "officially" support C++
- we had to develop our own "object oriented database" layer on top of the relational technology
- relational technology imposes a performance overhead for simple tasks like event logging
- version upgrades, from 6.2 to 6.4, caused hiccups in the development schedule

Managed Equipment

One of the main functional requirements for NMCS II was to support a new generation of T-1 equipment. This generation included a customer-programmable cross-connect (switch), a low end multiplexer, new logical objects for managing services, and a variety of new access equipment. The incorporation of these new features into the functional model implemented in NMCS-PC varied in difficulty. Some of the new equipment could be added by simple subclassing from existing superclasses. In the worst case, however, we had to define radically new objects and management algorithms to support user-level abstractions which were not correctly implemented in hardware.

Acceptance *vs.* Acceptability

The NMCS II system is a state-of-the-art application utilizing advanced hardware and software technologies and object oriented development methodologies. The delivered product has successfully met its stated requirements. However, the final test of the product is acceptance by its users – the *MEGA*Stream customers and the support organization which must support the product and the end users.

In evaluating NMCS II it is important to distinguish between the acceptance of the product and the acceptability of the product. *Acceptance* is a objective concept which can be precisely measured, and is defined as the ability of the product to meet the requirements imposed upon it and pass a formal acceptance test. Final Acceptance is a formal contract milestone between the supplier of the product (TCSI) and the customer (SRCI).

In contrast, *acceptability* is a subjective measure of the usefulness of the product to meet the users' requirements. In a perfect world, the users' requirements exactly match the formal product requirements, and acceptance and acceptability are equivalent. In the real world, however, it is impossible to precisely capture the users' requirements in a formal sense. This is due to a number of factors:

- Users are network managers or system administrators, not software developers. They understand their job in terms of their tasks and responsibilities, but cannot readily translate those tasks and responsibilities into functional requirements.
- Prototypes are useful but of restricted value. A prototype is useful for a user to evaluate look-and-feel issues and comment on new features, it does not allow him to test the product in an operational scenario.
- Due to schedule and budget restrictions, it is not always possible to fully satisfy the users' collective requirements in a first release.

Each of these points can be addressed to some degree using a variety of techniques such as focus groups, early involvement of users in product testing, and proper expectation management in the face of limited resources and phased delivery schedules.

The final obstacle to NMCS II acceptability was the existence of NMCS-PC. As described earlier NMCS II was to replace NMCS-PC functionally, as well as provide a new usage paradigm based on the new technology generations in the delivery platform. While the functionality of NMCS II was generally acceptable, the paradigm change created a significant gap in the perceived usability of the product. This "technology generation gap" manifests itself in three key areas:

- Usability
- Performance
- Deployment

The first two areas are of primary concern to the end-user community – the network operators and managers responsible for managing the network to support the business applications which use the network. The third area is of concern to the Telco operators, who are responsible for maintaining the NMCS II application and answering questions from the end-user community. For each of these areas we will describe some of the feedback we have received from the appropriate user community, and how we are addressing the acceptability issues raised by those users.

Usability

Usability can be defined loosely as the ease with which the user can do his job using NMCS II. Some early users were unhappy with the usability of the first release of the product; many of their complaints can be traced directly to the technology generation gap.

User Interface. NMCS-PC used a character-base user interface. It presented a limited amount of information on the screen at one time, and relied heavily on function keys to navigate between screens and invoke actions. NMCS II, in contrast, makes extensive use of windows and the mouse.

As we delivered the first release of NMCS II we expected a transition period during which users would have to familiarize themselves with the mouse-and-windows interface. Many of the problems the users reported indicated that the transition is going to require more time than we had anticipated; some of the problems or enhancement requests we heard from users included:

- there is too much information on the display because of the ability to open multiple windows
- data entry and operation invocation require the mouse; many users dislike the mouse and prefer function keys
- users prefer to operate from a tabular list of objects, rather than a detailed view of individual objects; this reflects the list-oriented behavior of NMCS-PC.

Taken as a whole, these concerns indicate that the technology change from character to graphical user interfaces can greatly impact the usability of a product, particularly if the users do not have prior exposure to graphical user interfaces. In follow-on release we have been adding features to NMCS II to make it less dependent on the mouse, and make it easier to work from lists of objects.

Functionality. Some of the usability problems users reported were due to changes in the way functionality was designed and implemented between the two products. We have received comments of the form "I don't like NMCS II because it does *XYZ* differently than NMCS-PC." This reflects a natural inertia on the part of the users, who become proficient with their current system and are reluctant to change the way they work.

Many of the changes to existing functionality were required to support features or functions introduced in NMCS II, and it would have been impossible to support both modes of operation in NMCS II. We are addressing this issue in future releases by involving users early in the testing cycle at TCSI offices in Berkeley. This approach provides a number of benefits:

- Users have an opportunity to test new features in the context of an operation product, not a prototype. If there are usability issues identified the development team can take steps to correct the problem before the product is delivered.
- Software developers and testers have an opportunity to learn how users work with the product in the field, which can increase our internal testing effectiveness.
- By socializing with the developers and testers, users become "part of the team," which opens the communication channels and provides a rich source of requirements and new ideas for follow-on development.

Performance

We define performance as the time taken to complete a given operation, and includes not only the performance of the application software, database system, and networks, but also the time taken by the user as part of the operation. Many of the performance issues which have gated product acceptability are due to the technology generation gap.

User Interface. Changing from a character-based user interface to a full X/Motif windowing interface has resulted in a noticeable degradation in responsiveness. The amount of tuning which can be achieved is limited by the complexity of the individual screens and the overhead of accessing Motif uid files.

Database. NMCS-PC utilized a custom database which was tuned for the application, but did not readily support queries from external processes. Switching to a commercial relational database in NCMS II (Ingres) allowed us to rely on the standard database capabilities like backup and recovery, to “open up” the data stored in the application to other applications, and to enable the use of SQL-based tools for custom reports and other queries. These capabilities come at the cost of performance, however, as even a highly tuned commercial RDBMS cannot compete with tuned in-core structures.

Functionality. There are several cases in which functions or features introduced in NMCS II have significant negative impacts on a user’s productivity. For instance, a major change in functionality from NCMS-PC to NMCS-II was the addition of element management capabilities for the multiplexers. This new function requires the creation of several hundred individual objects for each multiplexer. Although the time to save an object to the MIB is less than 5 seconds, the time to open the screen and enter the configuration data for each object is over 1 minute.

Performance Improvement. Most performance acceptability issues can be traced to comparisons to the NMCS-PC product. NMCS-PC is a mature application, it has been highly tuned over its lifetime, and users have adapted to the product and have found the most efficient ways to use it. The performance degradation inherent in the technology changes described above, coupled with the high performance standard set by NMCS-PC, virtually guaranteed that users would rate NMCS II slower than NMCS-PC.

We have addressed the performance concern in two key ways:

- Continued incremental performance improvements in all areas of the system, including software optimization, database tuning, hardware upgrades, and reconfiguration of the client-server environment to maximize throughput.
- Redesigning the tasks which operators perform with NMCS II to reduce the operator interaction. For instance, in a pending release we will provide a capability to partially provision a database by uploading information automatically from the network. While this operation takes considerably more time than creating an individual object, it can replace up to 150 individual operations, resulting in two orders-of-magnitude improvement in user productivity.

Recent efforts in these two areas have increased the overall system performance to acceptable levels, and we have committed to a program of continuous performance improvement throughout the NMCS II product lifecycle.

Deployment

Deployment issues include Operations, Administration and Maintenance, and are the primary concern of the Telco users. As with the other two areas – Usability and Performance – many of the acceptability issues arise from the technology generation gap.

Hardware Platform and Operating System. NMCS-PC, as its name implies, was delivered on an Intel-based personal computer running QNX, and it used VT100-compatible terminals over dial-up lines for remote access. In contrast, NMCS II is delivered on Sun SPARCStation servers, and uses SPARCStation workstations over LAN or dedicated links for remote access; both client and server platforms run the SunOS operating system. In our experience, and in the feedback we have received from the Telco users, maintaining networked UNIX workstations is much more difficult than the equivalent number of PCs, as it requires more staff with greater knowledge and experience. Many of the problems which the users have experienced in this area are part of the steep learning curve which is associated with converting to UNIX.

Application Architecture. NMCS II uses a distributed process and data architecture, with the SMIB/MMIB separation as a key architectural component. Each executable utilizes a number of libraries, utility programs, and configuration files to ensure proper operation. The GUI process requires several hundred auxiliary files which contain the screen definitions (Motif uid files), online help, and other configuration data. As a whole, an NMCS II distribution tape includes over 1500 files in 40 directories.

Given the complexity of the software distribution, we spent considerable time developing maintenance and packaging scripts to configure an application. These scripts were targeted to build, install and maintain a "standard configuration," which did not always match the particular needs of individual customers. Therefore, the operations staff has had to invest considerable time producing maintaining custom configurations.

Database. The Ingres relational database which NMCS II uses is itself a complex product, which requires specialized knowledge to install and maintain.

Deployment Improvement. Most of the issues listed above arise from the additional complexities introduced with the technology change. Were are approaching these issues with a multi-prong approach.

- Increase the level of expertise within the operations staff, through training (formal and on-the-job) and consultants.
- Formally address the requirements for supporting custom product configurations, and deliver enhanced installation and maintenance scripts with new releases.
- Simplify the operation, packaging and installation.

Lessons Learned

The NMCS II development and deployment effort has been a challenging and rewarding undertaking. Both SRCI and TCSI have learned some lessons as we have bridged the technology generation gap, which we summarize below.

Comparison to the "Old System". Whenever a new system is being built which replaces an existing system, the new system will always be evaluated against the performance and usability of the system it replaces. While this comparison may seem unfair to developers, especially if the new system provides greater functionality or implements a different way of performing current tasks, the subjective impressions resulting from this comparison become real issues which must be addressed.

In some cases these issues may be antithetical to the basic design of the new system; education of the users on the design and usage of the product are the best solution to overcome these issues. In other cases, compatibility modes can ease the transition to the new system. In either case, the requirements analysts of the new system should fully understand how users use the old system – what are their likes and dislikes – and plan to explicitly address those features in the design of the new system.

End User Involvement. We cannot overestimate the importance and value of involving the end users in all stages of the product development life-cycle. Typically users are consulted during the requirements phase, and again during the acceptance phase. We have found that keeping users involved through the development and early testing phases improves the quality of the delivered product, opens the communication channels between the development and user communities, and can address and resolve usability issues before they become problems.

End user involvement would have been particularly beneficial in evaluating the new usage paradigm as implemented in the Graphical User Interface. The change from text to graphical presentations, with the corresponding change from function keys to the mouse input, is the first hurdle users will encounter with the NMCS II product. Exposure to prototypes and early releases could have smoothed that transition. In addition, the choice of uid files to define screen layouts allows field personnel to modify screens to conform to user requirements, without requiring changes to application code.

Attention to Deployment. Products like NMCS II, which have a relatively small number of installed sites but are custom-configured for particular customer needs, require special attention to deployment issues. Focusing on packaging and installing a "reference configuration" is not adequate under these circumstances. That is, the requirements analysis and definition process should include the needs of custom configurations. Stated another way, the deployment staff are part of the user community and must be included in normal product definition and development life-cycle.

Expectation Management and Continuous Improvement. In retrospect, the goal of delivering the NMCS II product, with all existing NMCS-PC and new functionality, on the new technology base, with the same performance and usability as NMCS-PC, *in the first release*, was over-optimistic. A product of this complexity requires one or more iterations to work the kinks out. We have adopted a program of continuous improvements to NMCS II through the maintenance and upgrade program. Coupled with tighter management of customer expectations – and greater visibility of customers into the improvement process – have greatly increased customer satisfaction with the product.

As part of the process of continuous improvement, we have taken steps to shorten the development cycle for follow-on releases. Reducing the time between releases allows us to react quicker to user issues as they arise, and makes very visible to the user the improvements. We are shortening the development cycle through a combination of the following techniques:

- Reducing the feature content of incremental releases, deferring items of lesser importance to later releases.
- Including members of the development team part in the requirements analysis process. The developer is best able to propose simple functional alternatives which satisfy user requirements.
- Beginning requirements analysis in parallel with the development cycle. This pipelined approach allows us to further reduce the time between releases without squeezing the development cycle.

Conclusion

Upgrading an existing application to a new technology base has many inherent challenges above and beyond the software development *per se*. The technology generation gap can introduce acceptability issues for a product which may not be captured by the formal acceptance process, in areas like usability, performance, and deployment. The best solution to bridging the gap is to recognize its existence in the first place, determine how the gap will impact the user community, and engage the users in the solution process.

Acknowledgments

The authors would like to thank the entire NMCS II development team in SRCI and TCSI, particularly John Reid, Pierre Veilleux, Denise Semak, and Steve Gale (SRCI) and Ira Rotenberg (TCSI), for delivery a quality product in which we can all take pride. We also owe a debt or gratitude to our user community for their patience and feedback, especially Jerry Abrams, Luc Bourdages, Kevin Johnson, Tracy Kluczynski, Eloi Leroux, and Bruno Smith.

References and Additional Information

NMS/Core is a Trademark of Teknekron Communications Systems, Inc., and is now marketed under the name BaseWorX™ Object Services Package. BaseWorX is a trademark of AT&T.