# Lessons learned from 25 years of process improvement: The Rise and Fall of the NASA Software Engineering Laboratory

**Victor R. Basili*+, Frank E. McGarry•, Rose Pajerski*, Marvin V. Zelkowitz*+**
* Fraunhofer Center for Experimental Software Engineering, Maryland, College Park, Maryland
+ Dept. Computer Science & Inst. for Advanced Computer Studies, University of Maryland, College Park, Maryland
• Computer Sciences Corporation, Lanham, Maryland
basili@cs.umd.edu, fmcgarry@csc.com, pajerski@fc-md.umd.edu, marv@zelkowitz.org

## Abstract

For 25 years the NASA/GSFC Software Engineering Laboratory (SEL) has been a major resource in software process improvement activities. But due to a changing climate at NASA, agency reorganization, and budget cuts, the SEL has lost much of its impact. In this paper we describe the history of the SEL and give some lessons learned on what we did right, what we did wrong, and what others can learn from our experiences. We briefly describe the research that was conducted by the SEL, describe how we evolved our understanding of software process improvement, and provide a set of lessons learned and hypotheses that should enable future groups to learn from and improve on our quarter century of experiences.

## 1. Introduction

If one thinks of the Software Engineering Laboratory (SEL), the joint activity of NASA Goddard Space Flight Center (GSFC), the University of Maryland, and Computer Sciences Corporation, one may think of a well-known software development environment at the forefront of software process improvement activities and the first recipient of the IEEE-SEI Software Process Achievement Award in 1994. But all that is in the past. The SEL as we knew it is gone. In this paper we present a history of the 25-year lifetime of the SEL along with our lessons learned on what was done correctly and what was done incorrectly as our legacy to future software process improvement activities.

We present this overview from an historical perspective. In Section 2 we give an overview of the SEL from its inception in 1976 until today and describe the impact that it had. We briefly summarize the changes over this period in our understanding of software process improvement. The SEL went through three distinct phases, where each phase changed our view of process improvement from an early naïve view that was data-centric to a more mature view that addresses the needs of the organization as a driver for appropriate improvement activities.

In Sections 3 through 5 we present a discussion of each phase of SEL evolution: from its inception in 1976 through the early 1980s, from the mid-1980s until the mid-1990s, and then from 1995 until today. We describe each phase from the points of view of the activities carried out, the management of the SEL, funding issues, and research conducted. We offer a series of lessons learned from our quarter-century of experiences as well as a concluding series of points that we believe reflect some of the issues that future improvement groups need to address.


*ICSE'02,* May 19-25, 2002, Orlando, Florida, USA.


## 2. SEL Overview

In April of 1976 personnel from NASA/GSFC began discussions with faculty at the University of Maryland for a research program in software engineering. These discussions resulted in a University of Maryland grant in August 1976 between GSFC's Systems Development Section (the precursor to the Flight Dynamics Division) and the University of Maryland's (UMD) Computer Science Department.

The original goals of the SEL were "to analyze the software development process and the software produced in order to understand the development process, the software product itself, the effect of various 'improvements' on the process with respect to the methodology, and to develop quantitative measures that correlate well with intuitive notions of good software" [Basili77]. Thus the SEL's original objectives were to promote an understanding of

- the underlying principles involved in software and the development process,
- the factors that affect the development of software and their interrelationships,
- the characteristics of classes of problems and products: types of problems encountered and errors made in developing a particular class of products, whether or not a particular methodology helps in exposing or minimizing the number or effect of a class of errors, what the relationship is between methodology and management control.

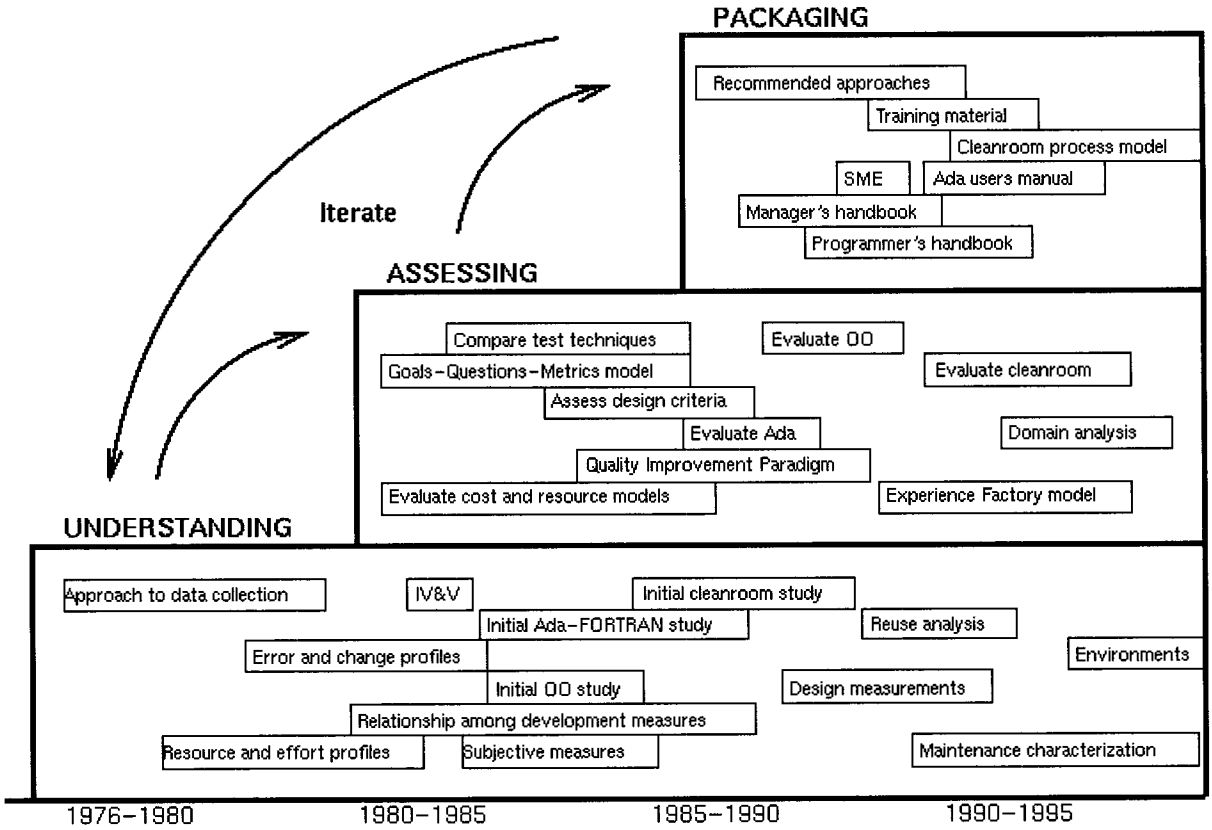


**Figure 1. Some SEL studies**

## 2.1 Research focus

From the beginning, the basic methodology of the SEL was to experiment by observing projects under development at NASA and collecting relevant data to be able to address the above objectives.

By viewing every project as a research activity, the research staff (principally from the University) was able to work with development personnel to improve their procedures. While a controlled experiment is desirable in order to understand the different impact from alternative models of development, the costs of replication are prohibitive. For this reason, the basic model was the case study.

In this model, the standard NASA approach was used to develop a project, with the research staff proposing minor modifications to that process. By running repeated case studies, each differing slightly, a large database of related projects were collected. Such projects are known as in vivo multi-project experiments since the impact on live projects that will be used on NASA missions is being studied.

From the University's perspective the SEL was not viewed as an isolated research activity, but instead was the center of an activity called the Experimental Software Engineering Research Group (ESEG). Many studies were funded by many organizations (e.g., government grants from Air Force Office of Scientific Research and National Science Foundation, and industrial sponsors such as IBM, and Burroughs Corporation, among others). Often these pure research studies were conducted as experiments using students at the University. If the experiment was successful, a more extensive experiment or case study was built around a SEL activity at NASA.

For some technologies that can be isolated in the laboratory, the University provided a good testbed. Various analytic strategies (e.g., structural testing, functional testing, Cleanroom, perspective-based reading) were first isolated and tested using university students. When these techniques proved effective in the laboratory, small projects at NASA were the next testbed. Gradually such processes were incorporated into the NASA environment and became part of the general software development process. Both types of testbeds are needed to test out ideas in the small, and then transition them into industrial practices.

## 2.2 Impact of the SEL

The many research accomplishments of the SEL have been reported elsewhere. We only briefly summarize them here. The SEL has been at the forefront in software process improvement

research. Some of the studies undertaken by the SEL are given in Figure 1. The SEL has contributed to knowledge of measurement, defect detection, IV&V, Ada, object-oriented design, COTS (Commercial off the shelf) software, as well as developing experimental methods for performing empirical software engineering research.

## 3. 1976-1983: Emphasis on data collection

Early SEL activities centered on data collection as a means to extract information from ongoing projects. The SEL personnel roster in early 1977 listed ten names: four from GSFC and six from UMD, none of whom would be fulltime on SEL activities. Initially, twelve ongoing development projects were to be monitored, measured, and analyzed to begin the work of understanding process and product within this one GSFC development environment. Seven data collection forms were designed to provide data on the projects' software characteristics (overall and at a component level), changes and errors during all phases of development, and effort and computer resources used. The data was collected at GSFC from NASA developers and the main development contractor, Computer Sciences Corporation (CSC). The data was then manually reviewed at GSFC before being sent to UMD for entry into the project measures database using a UNIX-based Ingres system.

From 1976 through 1978, SEL activities focused on validating, and analyzing the 2000 forms collected from project personnel in order to support experiments in the form of

- *screening* experiments (understand what's going on now) - baselining current GSFC projects,
- *semi-controlled* experiments - comparing different methodologies on different related projects,
- *controlled* experiments - replicating the same project using different methods at GSFC and UMD.

The naïve simplicity in which data was collected broke down by 1978 and a more rigorous set of processes were instituted. This could not be a part-time activity by faculty using undergraduate employees. *Lesson 1: Data collection requires a rigorous process and professional staff.* In addition, we had to compromise on the amount of data we wanted versus the amount of data that realistically could be collected. *Lesson 2: You must compromise in asking for only as much information as is feasible to obtain.* Forms were shortened to allow for more complete collection.

The data collection process for the 20 projects then under study became more rigorous with this 5-step approach:

1. Programmers and managers completed forms
2. Forms initially verified at CSC
3. Forms encoded for entry at GSFC
4. Encoded data checked by validation program at GSFC
5. Encoded data revalidated and entered into database at UMD. (After several years, CSC took over total management of the database.)

To obtain contractor cooperation, a 10% overhead cost to projects was allocated for data collection and processing. But this was rarely needed. These costs initially were about 5% and over time dropped to 1% to 2% of development costs. However, with the analysis function of the SEL added in, total costs remained about 10%. This included experimental analysis, measurement, and report generation.

### 3.1 Management

The SEL began as a relatively simple grant between two university faculty members and the Systems Development Section of GSFC. However the complexity of data collection and entry soon overwhelmed the university.

In 1978, the SEL partnership formally expanded to include CSC, the prime flight dynamics application development contractor. Since that time the SEL has always been a partnership of these 3 organizations: NASA/GSFC, UMD, and CSC.

We quickly learned that success required the cooperation of all three partners:

- NASA was needed to provide the management structure to see that the appropriate spacecraft software was developed and to ensure that the necessary data was collected. NASA provided the basic questions (i.e., the areas that needed improvement) that drove SEL research.
- The University was needed to oversee the research questions under study, provide research hypotheses to test, and analyze the data.
- CSC, the support contractor, was brought into the process to help with two major activities: process conformance and data collection.

Many of the day-to-day activities of the SEL involved the data capture activities of the developers. Getting their support was an important early step. At first there was considerable reluctance to cooperate:

- There was fear that the data collected would be used for personnel evaluation. We constantly assured the staff that this was not so.
- There was also a fear that data collection would cost too much. For example, when CSC initially stated that data collection would substantially increase the cost of software production, NASA's reply was that they would cover the cost of measurement and process improvement. CSC was told to take whatever time was necessary to process the forms. In the end the additional cost was only a couple of percent and we believe well justified for the results obtained.

During the early days in the late 1970s we ran many workshops training the staff to fill out the forms in a reasonably consistent manner. Because of staff turnover, this problem disappeared in several years. Later hires filled out the forms from day one and assumed it was part of their normal activities. However, there was a tendency to grow complacent and not be as thorough in training on data collection. *Lesson 3: Staff training in data collection is a never-ending vigil.*

In order to keep staff interest, there was a need to provide feedback to project managers. At first the complete cycle took several months from filling out forms, to data processing, to analysis. We were constantly working on methods to provide information back to project managers in a timely manner. We

finally developed the SEL library as a mechanism for providing a continuing series of reports on our activities. Also, when there was a critical time problem on some project, much like other organizations, data collection was suspended while the staff worked on other crises. *Lesson 4: As important as data collection is, it still takes second place to deadlines.* The data we collected was useful, but not always complete. Accounting for missing and incomplete data had to become part of our data collection and improvement process.

## 3.2 Research Funding

The SEL began as a one-year NASA grant (NSG5-123) to the University of Maryland for $50,000. This was extended to become the longest running grant at GSFC, ending in 1996 when new accounting procedures forced its replacement with a new grant. In 1976, $25,000 covered a faculty summer salary and a graduate student for a year (plus university overhead); at today's rates, this would cost about $90,000. Rates for CSC technical staff averaged $50,000 during the initial years and have almost tripled over the 25-year period.

Figure 2 shows the distribution of approximately $5.7M funding for the UMD over 25 years in both annual allocations and constant 1976-dollars. Beginning at $50K annually, the effective rate (inflation adjusted) was never more than 3 to 4 times that amount.

Several times we were asked by other researchers how they could share in the SEL's millions of research dollars. The impression of large annual grants was generated by our ability to capitalize on other funded research activities. However, funding for the University's share of SEL activities rarely exceeded $300K per year.

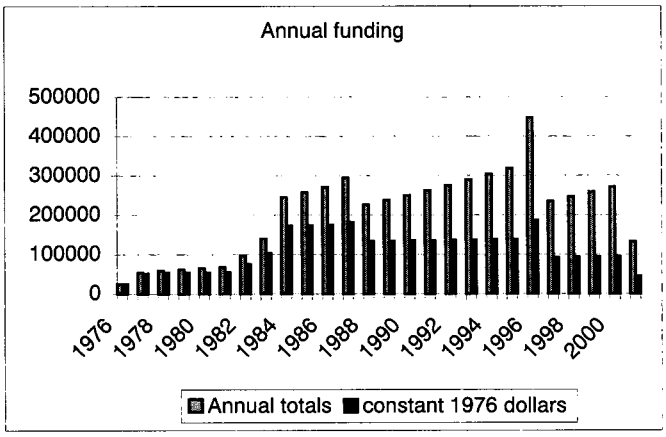


**Figure 2. UMD Funding**

## 3.3 Research

From the beginning the SEL was envisioned as a research organization, with research on process improvement being a major activity. It is just about impossible to count the number of papers that have been written by SEL personnel, but one measure is the number of papers that have appeared in the annual Software Engineering Workshops or have been cited in the set of 17 volumes of collected papers that appeared between 1981 and 2000.

A total of 265 papers and reports have been written. Of these, 92 presentations have been given at the Software Engineering Workshops from 1976 through 2000, and 38 were technical reports, most of which later appeared as conference and journal papers elsewhere. Figure 3 shows the distribution of papers over the lifetime of the SEL.

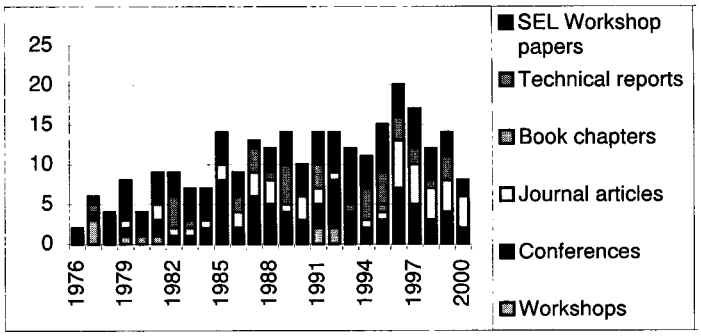


**Figure 3. Papers produced by SEL personnel**

As expected, there were few research results during the first few years. Most of the early activities centered on learning about data collection and developing our model of how to collect and process data. The first major paper was written in 1978 for the 3rd International Conference on Software Engineering, which was on resource estimation models using two early ground support software systems [Basili78]. From the mid-1980s onward, the SEL produced between 10 and 20 papers annually. The drop in the graph for 2000 is due a decrease in research findings as SEL activities have declined.

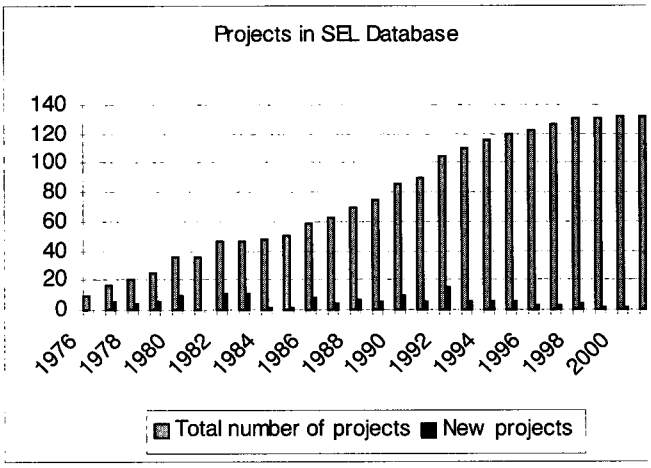


**Figure 4. Growth of SEL database**

Early research activities centered on learning how to collect data and run empirical experiments in the NASA environment:

**Data collection:** The initial focus of our research was on data collection processing. This was a much more complex process than we first imagined. The process evolved from the University processing the data collection forms, to CSC processing the forms and the University entering the data into the database, to CSC taking over the entire data processing activity. The growth of projects in the database are summarized by Figure 4.

**Experimentation:** Once we had data collection under control, our initial investigations focused on resource estimation and defect reduction techniques. The early work centered on trying to understand whether the data we were collecting fitted the models being proposed in the literature. We took ideas proposed by others (e.g., resource models such as the Rayleigh curve, and systems like SLIM and Price-S; testing models such as structural and functional testing) and investigated them in the SEL setting.

## 3.4 Lessons learned

During this period, lasting from 1976 until 1983, the SEL was concerned with learning about its environment. Our goals during these early years were to answer the following:

1. How do you conduct an experiment and learn from the efforts of building production-quality software?
2. How can you characterize the process being used to develop a software product?
3. How can you operate an organization like the SEL?
4. What does it cost to collect the information needed as you run an experiment?

Our initial list of observations in selecting the measures and collecting the data were:

- There was a 10% overhead to projects for data collection and analysis of the data by the SEL.
- No Hawthorne effect had been observed in the projects being studied. We feared that programmers under study would behave differently. However after several years, data collection became just one of the normal development activities, so this aspect of the work was no longer of concern.
- The SEL needed to provide feedback to projects quickly in order to keep developer interest in the activity.
- Subjective data was important in understanding the impact of a methodology.

From our early studies we were able to build models of the environment and develop profiles of the organization. These baselines and models provided the historical perspective that we needed to assess the impact of our experiments. *Lesson 5: Establishing a baseline of an organization's products, processes, and goals is critical to any improvement program.* We always had problems in obtaining accurate data from the developers, which led to the following lesson: *Lesson 6: The accuracy of the measurement data will always be suspect, but you have to learn to live with it and understand its limitations.* Similarly, immediate feedback to developers of the collected data is not possible, so you must make allowances for this need. *Lesson 7: There will always be tension between the need to rapidly feed back information to developers and the need to devote sufficient time to do an analysis of the collected data.*

Understanding how to use measurement developed into the Goal Question Metric (GQM) method in the late 1970s [Basili94a] and early 1980s. This process converted the data collection process from a random assortment of data objects into a tuned set of objects needed to address a stated goal.

# 4. 1984-1994: A Learning organization

By the early 1980s, we believed we understood how to measure a development environment, and our activities began to center on measuring the impact of changes by applying specific technologies (e.g., Ada, IV&V). However, we were drowning in data. We needed a method to decide what data was needed and how to collect it. Many experiments were run under an increasingly formal model of experimentation.

By the early 1980s the Quality Improvement Paradigm (QIP) was developed as a six-step method for applying the scientific method as a way to improve on the stated goals identified by the GQM process. By the late 1980s, systematically saving and reusing experience from previous projects became known as the Experience Factory approach (EF) [Basili94b]. Taken together, these three concepts (GQM, QIP, EF) provided a unifying framework for the SEL's experimental approach to software process improvement for future activities.

This unifying framework caused a rethinking in the long-range approach toward experimental research in the SEL. While the original concept given earlier centered on screening experiments, semi-controlled and then controlled experiments, the SEL adopted an abbreviated version of the QIP to represent its process improvement approach (as given in Figure 1 earlier). Our earlier emphasis on data collection and process characterization continued through the 1980s, but the emphasis on new activities focused on assessing specific solutions to certain technology problems and packaging solutions to them.

The 1980s can best be described as the period when we developed the concepts of how to build learning organizations. This evolved into the Experience Factory model. The three process improvement steps of the 6-step QIP describe the activities of characterizing, assessing, and packaging new technologies:

1. *Characterization* studies to understand the basic environment by collecting baseline data on an organization. Resource estimation, scheduling, defect occurrences, lines of code and other size measures were the early emphasis on our research. This was used to create baselines upon which we could study the impact of new technologies.

2. Later we moved into *assessing* the impact of various technologies. This involves goal generation, process selection, process execution, and analysis of the results, with a comparison to the earlier characterization studies. By the early 1980s we started to look at other technologies in order to see their impact within the NASA environment. The Department of Defense was proposing independent verification and validation (IV&V) and the Ada language as improvements to development. Both of these were investigated, as were Cleanroom and object-oriented techniques (OOT) and other technologies.

3. By the late 1980s, with the advent of the Experience Factory, we started to *package* technologies into useful chunks. For those analysis studies which demonstrated an improvement in development characteristics, guidelines, handbooks, training materials, or tool support were developed. Over time, a Measurement Guidebook [SEL94] was added. This Guide along with updated versions of the Manager's Handbook [SEL84], and the Recommended Approach [SEL81] became the printed version of our experience base and were widely distributed. Object oriented activities became encapsulated into the General Object Oriented Design document [SEL86]. The SEL had an active report publication series during this period.

So, the modus operandi for the SEL became that a process was adopted only after being applied on one or more pilot projects, shown effective in the environment, and evolved and packaged for use.

Major changes were being realized in flight dynamics software development from the mid-1980s through the mid-1990s. Functionality of the resulting systems increased 5-fold between 1976 and 1992. Table 1 briefly summarizes the SEL's accomplishments in several key product measures.

**Table 1. Impact of SEL research**

| Attribute | Change: 1987-1991 | Change: 1991-1995 |
|---|---|---|
| Decreased development defect rates | 75% | 37% |
| Reduced cost | 55% | 42% |
| Improved reuse | 300% | 8% |

The year 1994 was a banner year for the SEL. As a result of the baseline results, backed up by raw data on over 120 projects, and hundreds of papers, the SEL competed for and won the first IEEE Computer Society Award for Software Process Achievement. NASA and GSFC awards for the SEL partners followed [Basili95].

## Outreach

Outreach to the NASA community and elsewhere was extremely important. In 1984 we reported that the use of leading edge programming practices had increased productivity by 15%. All of this early knowledge was packaged into two core documents: the Recommended Approach to Software Development [SEL81] and the Manager's Handbook [SEL84], which have been revised several times since then. In contrast to other such documents, these were concise manuals, experience-based and routinely used. Since the head of the software development organization was also the head of the SEL, it was possible to enforce the use of SEL methods and support experimentation while ensuring that the research done had practical application. *Lesson 8: Having a shared commitment over research and development is vital for success.* This was crucial during the early years in making the SEL part of the development culture. Its lack in the late 1990s was instrumental in the eventual failure of the SEL in securing future projects to study.

Providing an outreach function to the community, an early activity was the Software Engineering Workshop, the first organized with 28 participants in late August 1976 to understand the current state of software engineering technology[1]. The SEL workshops took on a larger role during the 1980s as attendance grew steadily at these one day workshops The Workshops were important for external outreach, external analysis of results, learning about new trends and techniques, and as an incentive to the SEL staff that at the end of each year, results were to be reported and impacts assessed in a public forum.

The SEL celebrated its 15th anniversary in 1990 with its first 2-day workshop attended by over 500 researchers and practitioners. In a summation of SEL costs from 1975 through 1990, it was reported (in 1990 dollars) that the SEL had spent:

- $2.5M to UMD for research support
- $5.5M to CSC for research and technology transfer (this included overhead to development projects)
- $6.0M to CSC and others for data collection, validation, database support
- $130M for Flight Dynamics operational software (i.e., SEL costs were about 10% of development costs)

The 17th SEW in 1992 produced record attendance with over 600 people registering - a live broadcast had to be setup in another auditorium as the main NASA/GSFC auditorium seating was limited to 400 people. The next few workshops would top out at 450-500 registrants.

## 4.1 Management

By the mid-1980s, the SEL management structure evolved to include three components, all of which were crucial for success of the enterprise. This structure directed SEL activities for almost 20 years:

**NASA management:** NASA had operational control of the SEL. By the early 1980s we realized that this proved to be one of the most important aspects of the organization. The head of Flight Dynamics Division had a dual responsibility - manage the development of operational flight dynamics software that would be required for upcoming space missions, and manage the SEL and ensure that appropriate data was collected. This dual role ensured that the contractor adhered to the data collection schedule.

This is one of the ways that the SEL differed significantly from other process improvement organizations. Process improvement is often the role of the chief engineer or CIO office. In such organizations, the chief engineer can tell the staff to collect data, but such pronouncements are mostly ignored. In a study of 20 US and Japanese industrial organizations in the early 1980s [Zelkowitz84], we found that to be the case. It was also the model followed by NASA after the 1997 reorganization when project managers were empowered to make their own decisions.

**CSC as support contractor**: CSC was the primary developer of NASA flight dynamics software. They also saw an opportunity to enhance their processes by supporting studies paid for by

[1] There were few software engineering meetings at that time. The first software engineering conference was held in Washington in September 1975 and the second not until October 1976.

NASA. By being part of the SEL structure, CSC was kept aware of the goals for the SEL and the rationale for making certain decisions. This provided for a closer cooperation between the software developers and the SEL and allowed data collection to proceed in a timely manner. In addition, since CSC was responsible for building the database, they felt more comfortable in the way that the data was used. CSC also was involved in packaging and documenting the processes that developed in the SEL and transitioning new processes into everyday use.

**University of Maryland as researcher:** The University became the focus of the research activities for the SEL. New technology was often tested at the University, pilot studies were then carried out at NASA, and the University worked with CSC to transition the technology into practice at NASA [Zelkowitz96]. *Lesson 9: There is a symbiotic relationship between research and practice in software engineering, and both activities gain from the interaction.*

Because of the close proximity of the University to NASA, the University was able to interact on a daily basis with the development organization. By the mid-1980s, although developers didn't always understand the current goals of each research activity, the University was considered more than just an isolated research "ivory tower" and the frequent interactions between the University and the development groups meant that there was usually a cooperative development staff to work with. *Lesson 10: Close proximity of researcher to developer aids both.*

Although each of the 3 organizations had separate primary roles, each group was involved in all activities. NASA and CSC staffs were intermixed in software development efforts and all 3 groups participated in many of the research activities.

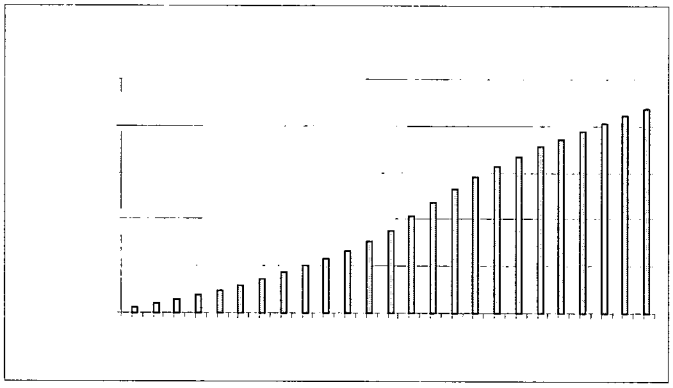

**Figure 5. Cumulative SEL funding**

## 4.2 Funding

Funding over this period was largely provided by NASA institutional funds (Code T and later Code O money), part of the development money provided for non-specific project support. Flight dynamics systems were considered general purpose in that they supported a number of different spacecraft. This funding was augmented by several smaller sources such as Code R (research) money that partially funded the SEL in the 1980s and Code Q (quality assurance and safety) money that provided funding in the early 1990s. A graph of cumulative funding is presented in Figure 5. SEL contractor and grant staff size during the mid-1980s started at about 7 full-time equivalents (FTEs) and gradually rose to 12-15 FTEs through the 1980s and peaked at about 20 FTEs in the early 1990s. NASA civil service staff that varied from 2 to 5 FTEs over this timeframe augmented these levels. This staffing represented from 4% to 20% of each project's development budget over the same period, still averaging 10% overall (as in the earlier 1976-1983 period). The large variation across projects was due to several factors:

- economies of scale - support cost for larger projects was reduced because the same infrastructure costs were incurred for staff sizes of 10-50 people.
- number of phases affected - the introduction of a specific new technology into one phase of the development process (e.g., a CASE tool for design) was less costly than one which impacted several phases of the lifecycle (e.g., OOD for design and code reuse).
- number of groups affected - while many of the studies impacted only the developers, some of the broader studies (e.g., GSS, the generalized software system) impacted several groups including the requirements specification analysts and the testers. These types of studies required more support resources for training and analysis.

## 4.3 Research

The driving force during this period was to make the SEL a learning organization. The goals for SEL research were to:

- liaison with the software engineering research community to identify potential technologies, which would solve specific development problems at NASA,
- evolve the SEL's empirical software engineering research program integrated within the SEL's flight dynamics development activities,
- build an experience base to feedback results from earlier SEL studies and provide insights to both future NASA missions and other software development groups outside of NASA.

The Goal Question Metric (GQM) method was developed around 1980 as a way to focus the SEL on what data was necessary to address certain perceived defects in the NASA development process. The Quality Improvement Paradigm (QIP) became the basic mechanism for feedback in the SEL so that results learned from one study could be applied on future projects. These evolved into the Experience Factory (EF) in the late 1980s as a learning organization model for understanding the development problems at NASA, characterizing the current environment, and developing and testing proposed alternatives to solve those problems.

As for specific technologies under study, reuse was becoming an important part of software development so additional studies to understand the characteristics of reusing software artifacts became an important part of the research during the early 1990s. Studies centered on Ada and OOT's impact as reuse rates grew from 20% to more than 80% on flight dynamics systems. CASE tools, domain analysis, and model building through machine learning (Optimized Set Reduction) techniques were being investigated; additional Cleanroom implementations were also

under study. A series of training courses were developed for use internally: these covered the SEL approach to process improvement, management through measurement, and flight dynamics fundamentals.

## 4.4 Lessons learned

The decade from 1984 through 1994 was the golden age of the SEL. The QIP, GQM, and EF frameworks were put into practice and an experimental paradigm for conducting empirical research lead to many accomplishments for the SEL. Several key ideas governed the SEL during this period:

- The foundation for improvement is the ability to produce a baseline of an environment. This gives an organization-centric view of what the current state of the environment is and where improvement is needed.
- Empirical learning is the model for improvement. Each project development is viewed as an "experiment" so that additional information can be incorporated into the experience base of the organization.
- Measurement is a required tool for process improvement. GQM focuses the organization on the goals and the product under development. It more clearly identifies what can and cannot be done and at what cost.
- Product measures rather than process changes must be the defined measures of improvement.

# 5. 1995-2001: Change

The elation of 1994 was tempered by concerns about budget cuts and reorganization plans for GSFC. From 1989 through 1994, SEL funding had been growing at an annual rate of 10% but that was ending and a 25-30% reduction was forecast. A NASA-wide reorganization was being planned and although it took several years before being put into place, rumors and concerns affected most employees. Amidst the GSFC reorganization rumors, a number of senior managers retired, including the Flight Dynamics Division Chief, a staunch SEL supporter.

New performance-based contracts were being formed and NASA managers no longer had the same insight into contractor practices. In 1994 the SEL lost its common management. New software development management did not have the SEL overview knowledge and the new SEL manager did not have the same control over developers that early SEL management had the previous 20 years.

The mantra of "faster, better, cheaper" became the word at NASA. New projects would rely more on outsourcing and COTS software. For NASA, using COTS was a revolutionary change to the standard custom system development approach. New processes needed to be developed on the fly and there was no time for "learning." The SEL was losing its contact with projects because of an increasing preoccupation on decreasing delivery schedules and costs by project managers. Therefore, the SEL was having difficulty in finding new projects that wanted its help. In response to this, the SEL began to study the effects of this new approach and evolve a process for its application. As a result, the SEL became less the driver and more the observer, less proactive and more reactive, to software development activities in its domain. *Lesson 11: Having upper management support is important for continued success.*

However, during 1995 and 1996, studies continued on COTS usage, requirements reading techniques, and OO architectures. The resultant process improvements continued to benefit the division's bottom line as costs decreased by an additional 10% and schedules were shortened by 5-20%.

At the end of 1997, the proposed reorganization was put in place, and a radically different organizational structure was implemented at GSFC. NASA mission teams were formed and developers were matrixed into these teams. Each team was empowered to decide what process it would follow and how it would report.

## 5.1 Management

For 20 years the SEL was part of the Flight Dynamics Division. However, under the reorganization, flight dynamics was no longer a separate division. (This was partially due to the SEL's success in reducing the complexity of building flight dynamics software.) After the 1997 reorganization, the SEL was placed in the Information Systems Center (ISC) of GSFC, whereas project managers were part of other NASA organizations. Under the new policy of empowerment, each project manager had control over the processes to be used in development of each project. ISC had to compete with outside contractors for continued development work. Cutting overhead costs drove major ISC decisions. The SEL lost the line authority it previously enjoyed over project development; the SEL began to lose touch with the developers in these mission teams; and the number of new projects being supported by the SEL decreased significantly. The SEL began to lose its ability to monitor and analyze projects.

As the new NASA settled into place, infrastructure organizations were dismantled if they couldn't find their own customer funding. For the past 2 years, the SEL has continued to seek out projects to work with that have an interest in reuse architectures or COTS integration, maintenance, and testing. *Lesson 12: The organization trying to improve their process has to own the improvement process.* Without the buy-in from project management, there was a loss of direction of what the SEL should do.

Management oversight of the SEL was vested in six directors, two from each organization. As a sign of impending problems, some of the director positions were given to individuals as honorary positions, rather than their knowledge or concern of SEL activities. There was no development manager working on SEL activities and initially no full time NASA SEL manager. Although eventually a SEL manager was appointed, he had no GSFC SEL staff to work with and little influence over development groups. In effect, there was no oversight or direction of the SEL after 1997.

Although UMD and CSC carried out most SEL activities, NASA personnel had a strong influence over SEL management. Who was assigned to the SEL became increasingly important,

especially after the management responsibilities for project oversight and process improvement were divided in 1994. During the height of SEL activities (e.g., during the 1980s) it was generally possible to maintain a quality research staff at NASA to work jointly with the University research team. However, by the mid-1990s, with the loss of top management support, maintaining a viable government staff was difficult.

NASA views itself as promoting space science research. It also views itself as primarily an engineering organization that "bends metal." Activities like the SEL in promoting software research in process improvement were not deemed to be critical technologies for GSFC, even though a large percentage of GSFC employees are computer professionals. Thus working with the SEL soon was equated to a dead-end position. More than one qualified government scientist refused a position with the SEL since there was "no place to advance to." This is a common complaint of many software professionals who work for engineering enterprises. *Lesson 13: It is difficult to make an engineering organization aware of the importance of software engineering to their mission.*

## Interaction with other GSFC groups

We were unaware of the importance of maintaining contacts with NASA personnel outside of our flight dynamics domain. The impact of this failing became quite clear after the 1997 reorganization.

**Upper management support:** Although the ISC development groups wanted to support SEL activities, considerable support was needed from upper management to make this happen. The SEL budget had been protected for almost 20 years during lean times since management saw the value of process improvement in lowering the costs and increasing the dependability of NASA software.

**Interaction with other NASA/GSFC organizations:** Due to the way GSFC evolved, Flight Dynamics had the responsibility for development of mission software, whether developed by NASA personnel or contractors assigned to work at GSFC. On the other hand, the Software Assurance Technology Center (SATC) had the role of providing quality assurance for software that was outsourced to other contractors. The SEL and SATC had very different processes and measures for quality. This was due to their different perspectives: development support vs. product assurance (external auditor). Because of their different roles, these organizations did not work together until 1997, when the need for a common measurement strategy for the mix of outsourcing and development was recognized. Had this happened earlier, the SEL might have had broader insight into applications outside of flight dynamics and been better positioned for the new NASA.

## 5.2 Funding

Beginning in the mid-1990s funding came from SOMO (Space Operations Mission Office). This office has as its mission "Provide space operations services that are responsive to customer missions at the lowest cost to the Agency." SOMO needed to be responsive to the needs of its customers who weren't convinced of the SEL's benefit. Funding for process improvement activities like the SEL didn't survive, and funding which was severely curtailed in 2001is for the near term ending in 2002.

## 5.3 Research

As NASA reorganized, our emphasis increased on reading technologies for understanding requirements and designs and on using COTS for mission software production. We built early baselines for the new ISC organization to understand it and characterize its domain problems and processes in the new heterogeneous environment. We characterized COTS work. COTS development is different since the underlying architecture of the COTS code already exists, so an understanding of the existing software architecture (e.g., reading the COTS specifications for understanding) must exist during the early specification phase of a project [Morisio00].

*Development* was the principal problem from 1976 through the early 1990s, whereas *building systems from components* was now the driving force. NASA managers had come to view the SEL as an organization that merely collected data on development activities. They were unable to see the SEL as a problem solving learning organization. This perception of the SEL made it difficult to find new projects with which to work.

## 5.4 Lessons learned

The period since 1995 can best be described as retrenchment. The domain changed from a homogeneous set of applications developed in the FDD to any number of applications in the ISC. New baselines had to be built for the new applications. Project managers were not willing to expend the resources to build new models. Multiple application domains also meant we had to deal with multiple group managers. We did not have the resources for such interaction.

We lost support of management. They did not understand what we did and we were not able to offer immediate solutions without understanding the environment. We lost the ability to interact with projects. Each project manager was empowered to do his or her own thing. They had no history with the SEL; did not understand what the SEL did; and saw the SEL activities of characterization and assessment as a potential overhead to their project.

Some of the observations coming from this period were as follows:

- There was no owner of the development process. Each project could have its own structure. The SEL no longer had a central model to build on.
- There was a loss of commitment on NASA/GSFC's part that process improvement was important. There was no vision on what was needed.
- There was significant restructuring at NASA/GSFC that the SEL could not keep up with.
- On the SEL's part, there was poor salesmanship in selling our vision to NASA project managers.

In the end, NASA management did not see the value of the SEL. For a while NASA seemed to stop directing the consortium, but just continued to give UMD and CSC money to run it. CSC, not seeing a continued commitment on NASA's part to fund the SEL, could not put additional resources on its own to keep it going. The University did not have the resources or expertise to run a development organization. The SEL quietly faded away toward the end of 2001.

## 6. Conclusions

The NASA/GSFC Software Engineering Laboratory had a successful 25-year run of researching new technology for improving the software development process at NASA. As a result of the activity, we believe we have greatly contributed to the knowledge of software development.

But all consortium members made mistakes, so in the end the SEL as created in 1976 ceased to exist. The SEL was slow to seek other non-flight dynamics activities to extend the SEL's reach at NASA, feedback to users was often late or lacking, and in the end NASA management from the post-1997 reorganization never understood what the SEL was all about and what its value was. In the post-1997 reorganization, the SEL did not have the upper-management support to survive the new development paradigm. In this report we have outlined some of the issues that we believe were most important in both the success and failure of the SEL.

We believe the empirical model, based upon the frameworks of QIP, GQM and EF to build an experimental science of software engineering, is the right approach. However, we listed 13 lessons whose impact we could not fully address at various times in the life of the SEL. Solving these should greatly aid future process improvement activities.

These lessons can be grouped according to several categories:

**Need for collecting project data:**

Lesson 1: Data collection requires a rigorous process and professional staff.

Lesson 2: You must compromise in asking for only as much information as is feasible to obtain.

Lesson 5: Establishing a baseline of an organization's products, processes, and goals is critical to any improvement program.

Lesson 6: The accuracy of the measurement data will always be suspect, but you have to learn to live with it and understand its limitations.

Success comes from integrating the basic process improvement steps (plan to experiment, measure, analyze data, ...) into the development process. This combination made improvement part of "how we do it" and just part of the culture. Using case studies within a narrow domain to simulate controlled studies works in a relatively homogeneous environment like the Flight Dynamics Division.

**Need for management buy-in to the process:**

Lesson 8: Having a shared commitment over research and development is vital for success.

Lesson 11: Having upper management support is important for continued success.

Lesson 12: The organization trying to improve their process has to own the improvement process.

Lesson 13: It is difficult to make an engineering organization aware of the importance of software engineering to their mission.

Project management and data collection must be under common control. Without a mandate to collect the necessary data, project personnel quickly lose interest. Other activities (e.g., a late project) take precedence and the data loses. The growth of online data capture on the web should lessen the problems we had in the early 1980s.

While you'll never get 100% management support at any level, to maintain the improvement program you need at least 25% of the first level managers in order to survive. So target one of the first outputs of the improvement program at that group.

**Need for a focused research agenda:**

Lesson 9: There is a symbiotic relationship between research and practice in software engineering and both activities gain from the interaction.

Lesson 10. Close proximity of researcher to developer aids both.

Using an experimental approach toward the problems of process improvement is vital. The program needs a research link - a university, corporate research group, or advanced technology consultant. On one level, it gives a development group access to emerging techniques. On another deeper level, the researchers are there to see that results are understood and measured, objective and subjective data are collected, the effects of scope are considered, and feedback is given to the participants.

Use multiple funding sources to best advantage. The SEL used research sources like the National Science Foundation to perform smaller replicated experiments, and if the results were promising, then SEL projects could be tried at NASA.

**Need for continued staff support:**

Lesson 3: Staff training in data collection is a never-ending vigil.

Lesson 4: As important as data collection is, it still takes second place to deadlines.

Lesson 7: There will always be tension between the need to rapidly feed back information to developers and the need to devote sufficient time to do an analysis of the collected data.

Establish an experience base as something you can touch, see, and use. The SEL Library was an effective demonstration of the SEL at work for both NASA developers and visitors. What does this mean in the age of online knowledge bases, the web, and global corporate intranets?

We don't have all the answers. And this report is not meant as a memorial to the SEL. What we hope we have done is given the reader some insights in how to do practical software engineering

research within an organization that builds large software systems. We have extracted 13 lessons from our 25 years of experience that should aid you in avoiding some of our pitfalls. Some of our conclusions may even seem contradictory. But that is the nature of the problem and why its solution is so hard. Ignoring the impossible will not make the problem go away, but by addressing it, you may see a way around it that simply escaped us.

## 7. Acknowledgements

This project was sponsored in part by the remaining funds in NASA grant NCC 5-464 to the University of Maryland. We also would like to acknowledge the hundreds of individuals who helped in the success of the SEL: programmers, analysts, and support staff at CSC and GSFC and the many students and visiting professionals at UMD who made the SEL the success it was. Space limitations prevent listing them all. We would also like to acknowledge the long-running support from NASA/GSFC and the NASA/IVV Center in Fairmont, WV for their support over these many years.

## 8. References

[Basili77] Basili V. R., Zelkowitz M. V., et al., The Software Engineering Laboratory, Technical Report TR-535, Computer Science, University of MD, May, 1977.

[Basili78] Basili V. R. and M. V. Zelkowitz, Analyzing medium scale software development, Third International Conf. On Software Engineering, Atlanta, Ga. (May, 1978) 116-123.

[Basili94a] V. Basili, G. Caldiera and D. Rombach, The Goal Question Metric Approach. Encyclopedia of Software Engineering. Wiley 1994.

[Basili94b] V. Basili, G. Caldiera and D. Rombach, The Experience Factory. Encyclopedia of Software Engineering. Wiley 1994.

[Basili95] Basili V., M. Zelkowitz, F. McGarry, J. Page, S. Waligora, and R. Pajerski, SEL's software process-improvement program, *IEEE Software* 12, 6 (1995) 83-87.

[Morisio00] Morisio M., C. Seaman, A. Parra, V. Basili, and S. Condon, Investigating and Improving a COTS-Based Software Development Process, 22nd International Conference on Software Engineering, Limerick, Ireland, 2000.

[SEL81] Recommended approach to software development, Software Engineering Laboratory Series, GSFC, April, 1983

[SEL84] Manager's Handbook for software development, Software Engineering Laboratory Series, GSFC, 84-101, November, 1990

[SEL86] General Object-oriented Software Development, Software Engineering Laboratory Series, GSFC, 86-002, August 1986

[SEL94] Software Measurement Guidebook, Software Engineering Laboratory Series, GSFC, 94-002, August 1986

[Zelkowitz84] Zelkowitz M. V., Yeh R. T., Hamlet R. G., Gannon J. D., Basili V. R., Software engineering practices in the United States and Japan, *IEEE Computer* 17, 6 (1984) 57-66

[Zelkowitz96] Zelkowitz M. V., Software Engineering technology infusion within NASA, *IEEE Trans. on Eng. Mgmt.* 43, 3 (August, 1996) 250-261