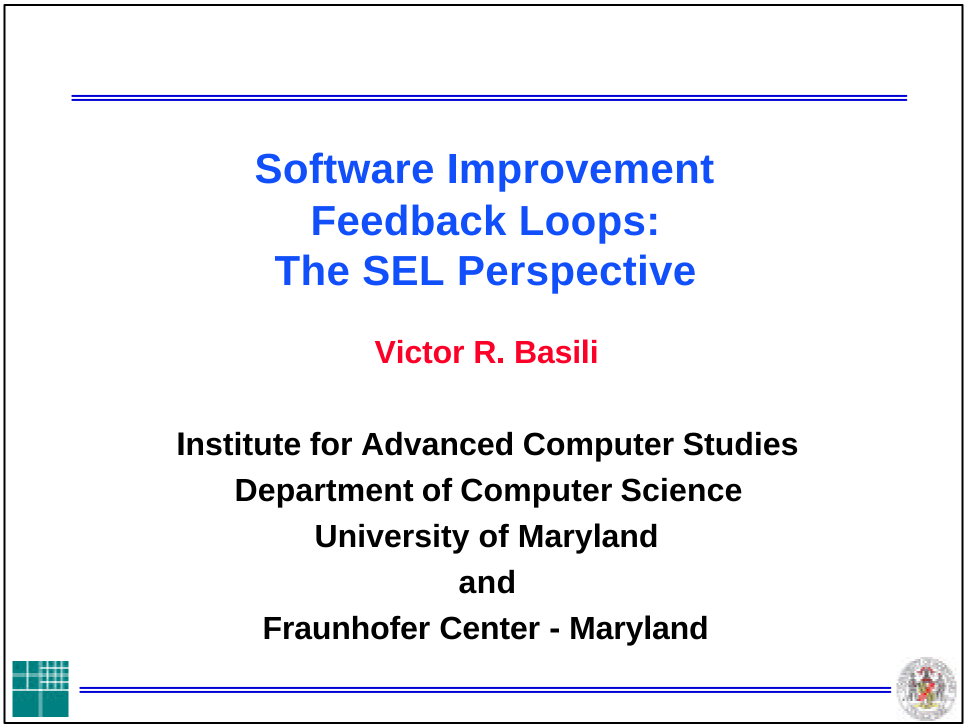
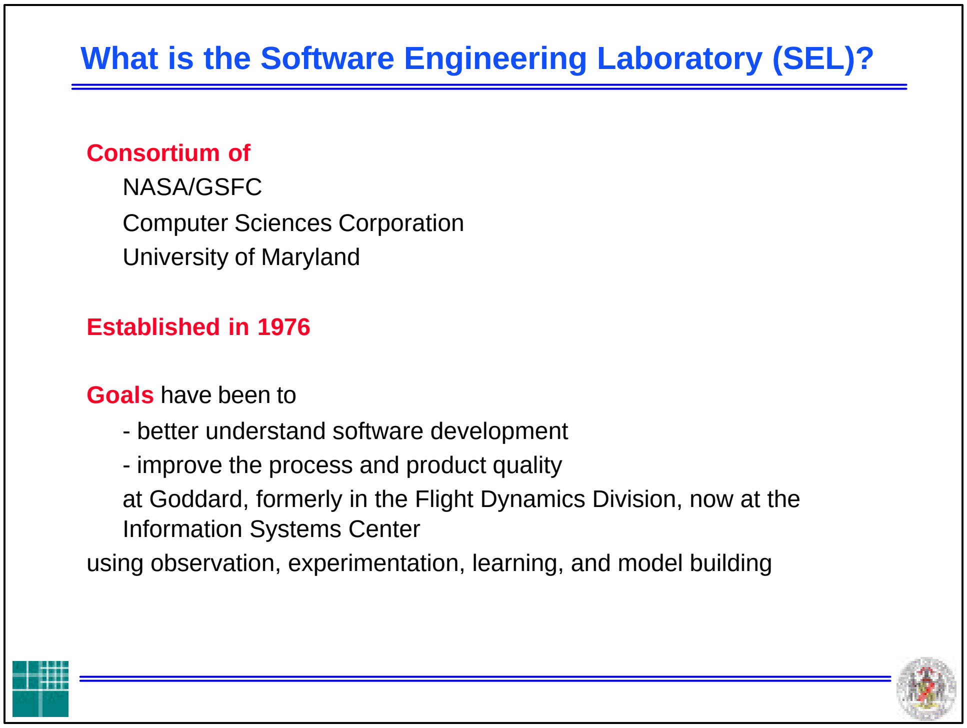


Software Improvement Feedback Loops: The SEL Perspective

Victor R. Basili

Institute for Advanced Computer Studies
Department of Computer Science
University of Maryland
and
Fraunhofer Center - Maryland



What is the Software Engineering Laboratory (SEL)?

Consortium of

NASA/GSFC
Computer Sciences Corporation
University of Maryland

Established in 1976

Goals have been to

- better understand software development
- improve the process and product quality

at Goddard, formerly in the Flight Dynamics Division, now at the Information Systems Center

using observation, experimentation, learning, and model building

Observation, Feedback, Learning, Packaging

Since 1976 we have **learned a great deal**, e.g.,
 understand before you assess
 data should be goal and model driven

Observation played a key role

Feedback loops have provided an environment for **learning**

Generated **lessons learned** that have been **packaged** into our process, product and organizational structure

Used the **SEL** as a **laboratory to build models, test hypotheses**,

Used the **University** to **test high risk ideas**

Developed technologies, methods and theories when necessary

Learned what worked and didn't work, applied ideas when applicable

Kept the business going with an aim at improvement, learning

This talk offers a retrospective and a look at our directions

Observation, Feedback, Learning, Packaging

The Quality Improvement Paradigm

The SEL
 1976 -1980 (Goal Question Metric Paradigm)
 1981 - 1985 (Baselining and Experimenting)
 1986 - 1990 (Experience Factory Organization)
 1991 - 1995 (Effects)

The SEL and Fraunhofer Center for Experimental Software Engineering
 1996-present

Quality Improvement Paradigm

Characterize the current project and its environment with respect to the appropriate models and metrics

Set quantifiable **goals** for project and corporate success and improvement

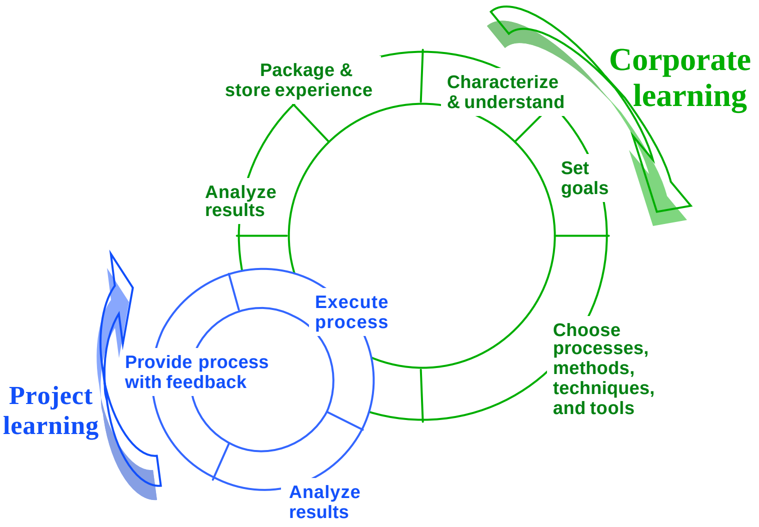
Choose the appropriate project **processes**, supporting methods and tools

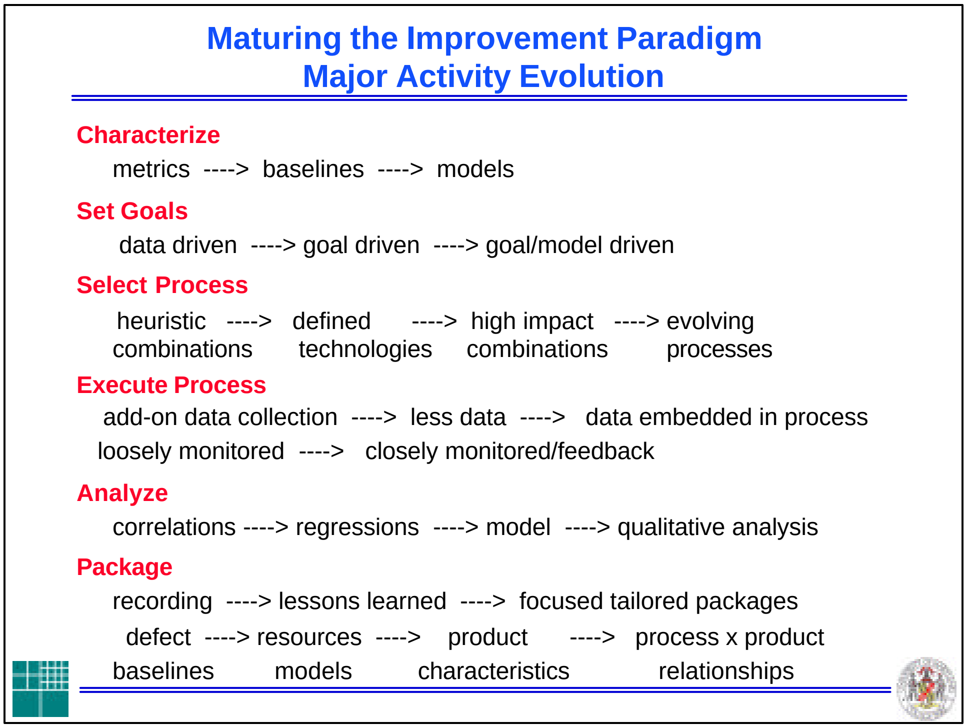
Execute the **processes,** construct the products, collect, validate and analyze the data to provide real-time feedback for corrective action

Analyze the **data** to evaluate current practices, determine problems, record findings, recommend improvements for future project

Package the **experience** in the form of updated and refined models and save it in an experience base to be reused on future projects.

Quality Improvement Paradigm


Corporate learning
Package & store experience
Characterize & understand
Set goals
Analyze results
Choose processes, methods, techniques, and tools
Execute process
Provide process with feedback
Project learning
Analyze results



Maturing the Improvement Paradigm
Major Activity Evolution

Characterize

metrics ----> baselines ----> models

Set Goals

data driven ----> goal driven ----> goal/model driven

Select Process

heuristic combinations ----> defined technologies ----> high impact combinations ----> evolving processes

Execute Process

add-on data collection ----> less data ----> data embedded in process

loosely monitored ----> closely monitored/feedback

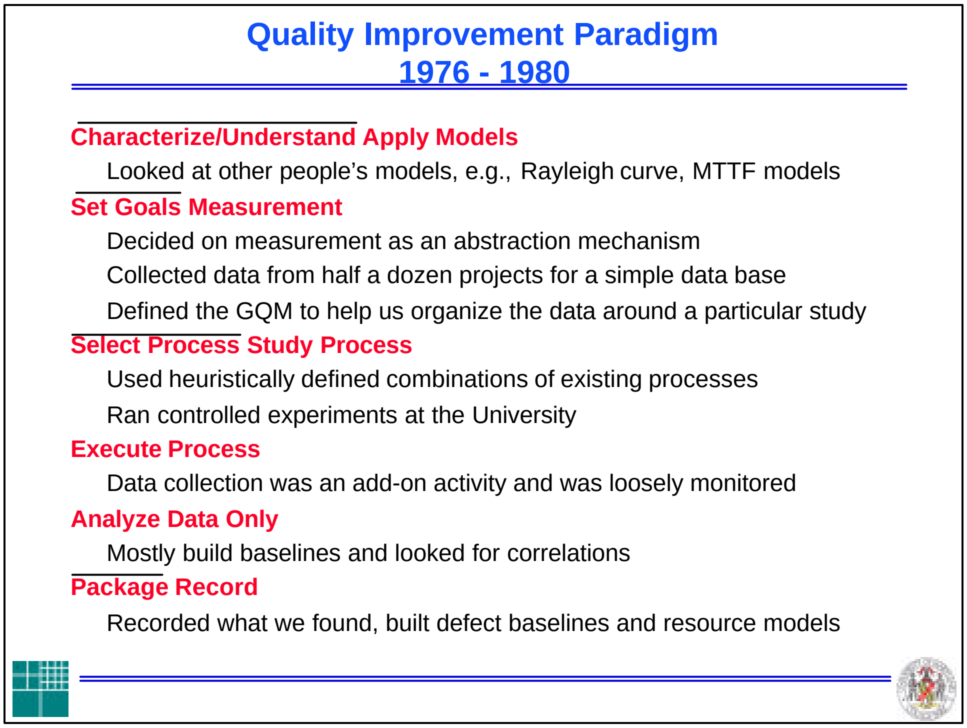
Analyze

correlations ----> regressions ----> model ----> qualitative analysis

Package

recording ----> lessons learned ----> focused tailored packages

defect baselines ----> resources models ----> product characteristics ----> process x product relationships



Quality Improvement Paradigm
1976 - 1980

~~Characterize/Understand~~ Apply Models

Looked at other people’s models, e.g., Rayleigh curve, MTTF models

~~Set Goals~~ Measurement

Decided on measurement as an abstraction mechanism

Collected data from half a dozen projects for a simple data base

Defined the GQM to help us organize the data around a particular study

~~Select Process~~ Study Process

Used heuristically defined combinations of existing processes

Ran controlled experiments at the University

Execute Process

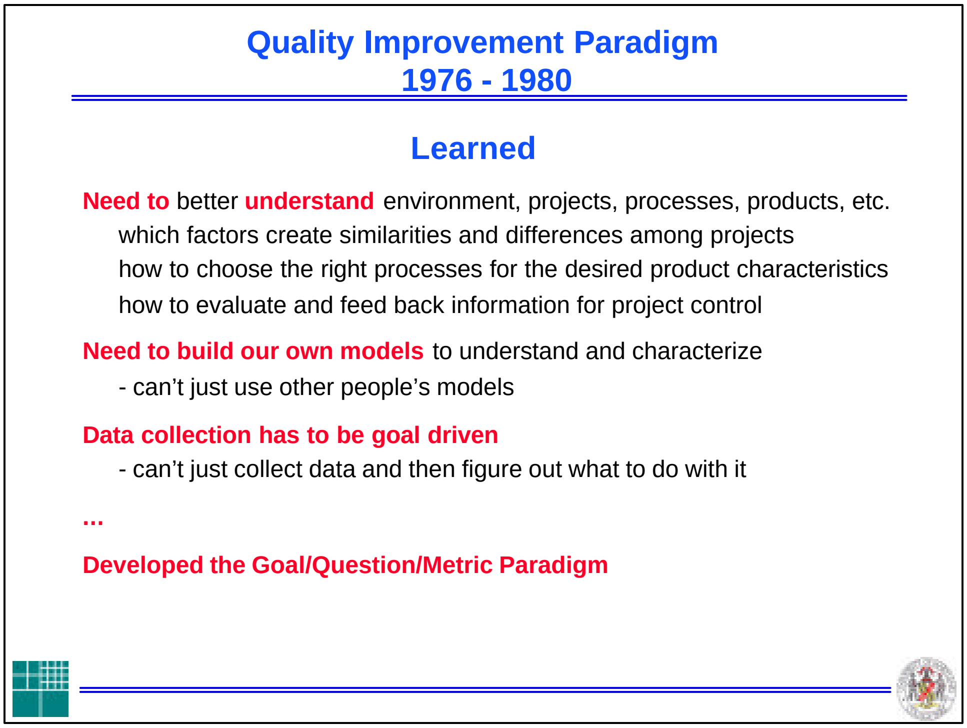
Data collection was an add-on activity and was loosely monitored

Analyze Data Only

Mostly build baselines and looked for correlations

~~Package~~ Record

Recorded what we found, built defect baselines and resource models



Quality Improvement Paradigm 1976 - 1980

Learned

Need to better **understand** environment, projects, processes, products, etc.

- which factors create similarities and differences among projects
- how to choose the right processes for the desired product characteristics
- how to evaluate and feed back information for project control

Need to build our own models to understand and characterize

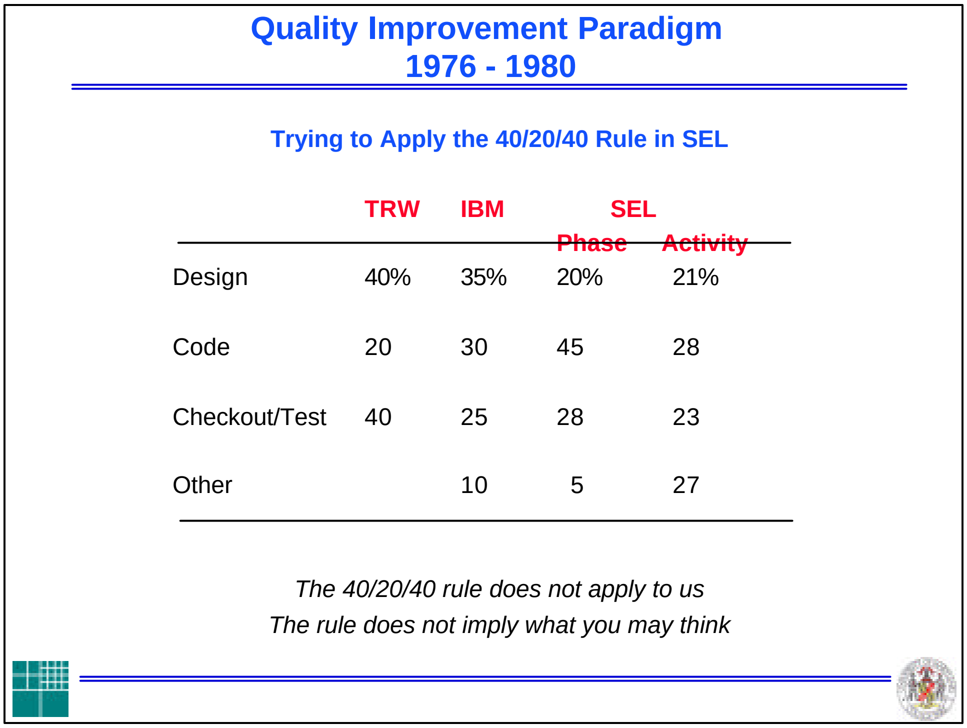
- can't just use other people's models

Data collection has to be goal driven

- can't just collect data and then figure out what to do with it

...

Developed the Goal/Question/Metric Paradigm



Quality Improvement Paradigm 1976 - 1980

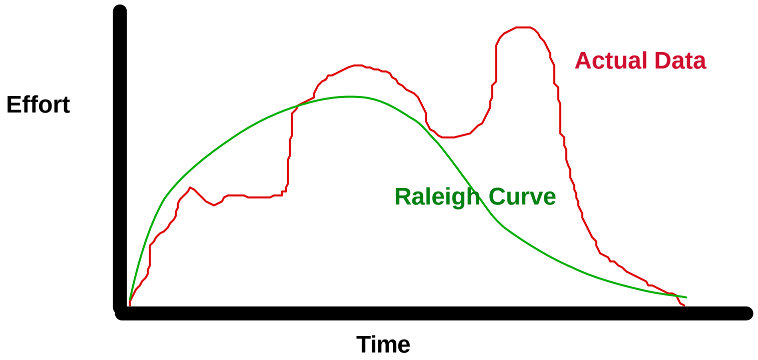
Trying to Apply the 40/20/40 Rule in SEL

	TRW	IBM	SEL	
			Phase	Activity
Design	40%	35%	20%	21%
Code	20	30	45	28
Checkout/Test	40	25	28	23
Other		10	5	27

The 40/20/40 rule does not apply to us
The rule does not imply what you may think

Quality Improvement Paradigm
1976 - 1980

Applying a Resource Allocation Model



Need to understand the local context
Local context makes a big difference

Quality Improvement Paradigm
Goal/Question/Metric Paradigm

A mechanism for defining and interpreting operational, measurable goals

It uses four parameters:

a model of an **object of study,**
e.g., a process, product, or any other experience model

a model of one or more **focuses,**
e.g., models that view the object of study for particular characteristics

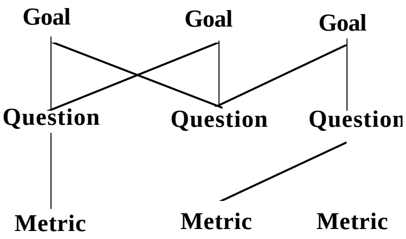
a **point of view,**
e.g., the perspective of the person needing the information

a **purpose,**
e.g., how the results will be used

to generate a **GQM model**

relative to a **particular environment**

GOAL/QUESTION/METRIC PARADIGM
Goal and Model Based Measurement



A Goal links two models: a model of the **object of interest** and a model of the **focus** to develop an integrated GQM model

Goal: Analyze the **final product** to **characterize** it with respect to the **various defect classes** from the point of view of the **organization**

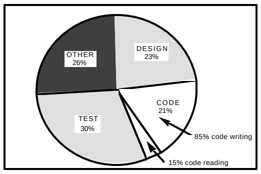
Question: What is the error distribution by phase of entry?

Metric: Number of Requirements Errors, Number of Design Errors, ...

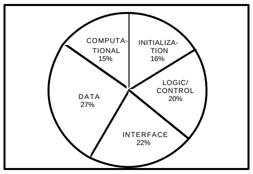
The Goal/Question/Metric Paradigm
Creating Baselines

NASA/SEL PROCESS BASELINE EXAMPLE

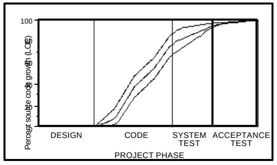
Effort Distribution*



Classes of Errors*



Source Code Growth Rate*

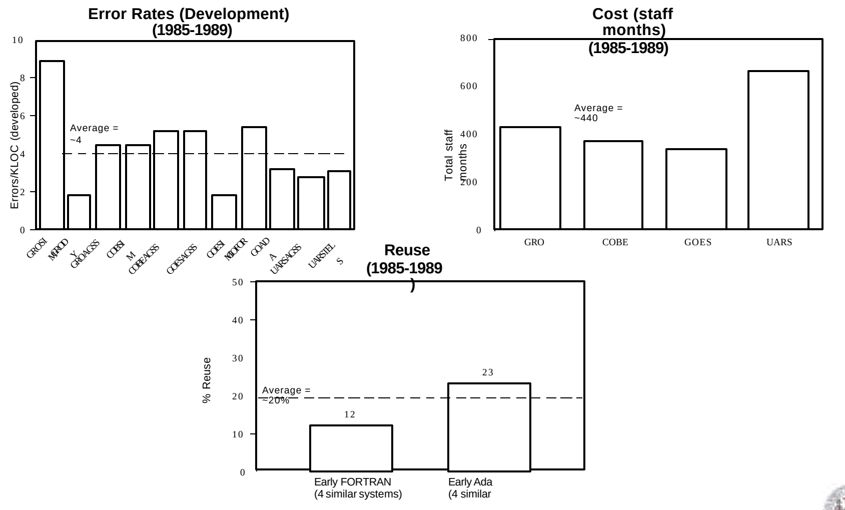


*Data from 11 Flight Dynamics projects (mid 1980s)

93M11U1.007

The Goal/Question/Metric Paradigm
Creating Baselines

NASA/SEL Product Baseline Example



Quality Improvement Paradigm
1981 - 1985

Characterize/Understand

Built our own baselines/models of cost, defects, process, etc.

Set Goals

Set GQM goals to study multiple areas

Incorporated subjective metrics

Select Process

Experimented with well defined technologies, e.g., Ada & OOD

Execute Process

Combine experiments and case studies

Collected less data

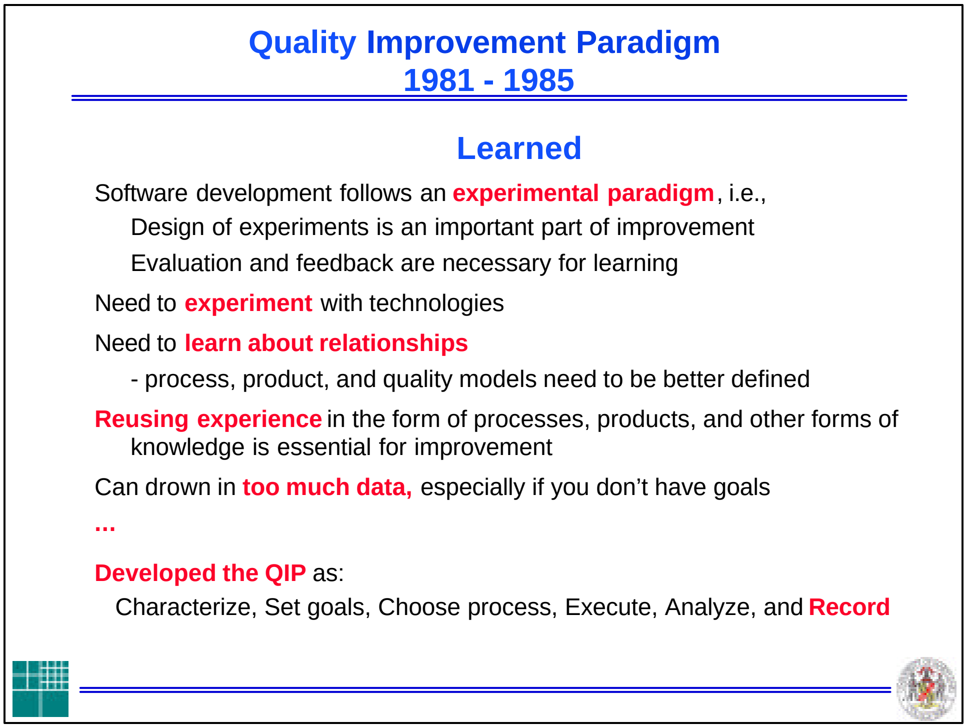
Analyze

Emphasis on process and its relation to product characteristics

Package Record

Recorded lessons learned

Formalize process, product, knowledge and quality models



Quality Improvement Paradigm 1981 - 1985

Learned

Software development follows an **experimental paradigm**, i.e.,

- Design of experiments is an important part of improvement
- Evaluation and feedback are necessary for learning

Need to **experiment** with technologies

Need to **learn about relationships**

- process, product, and quality models need to be better defined

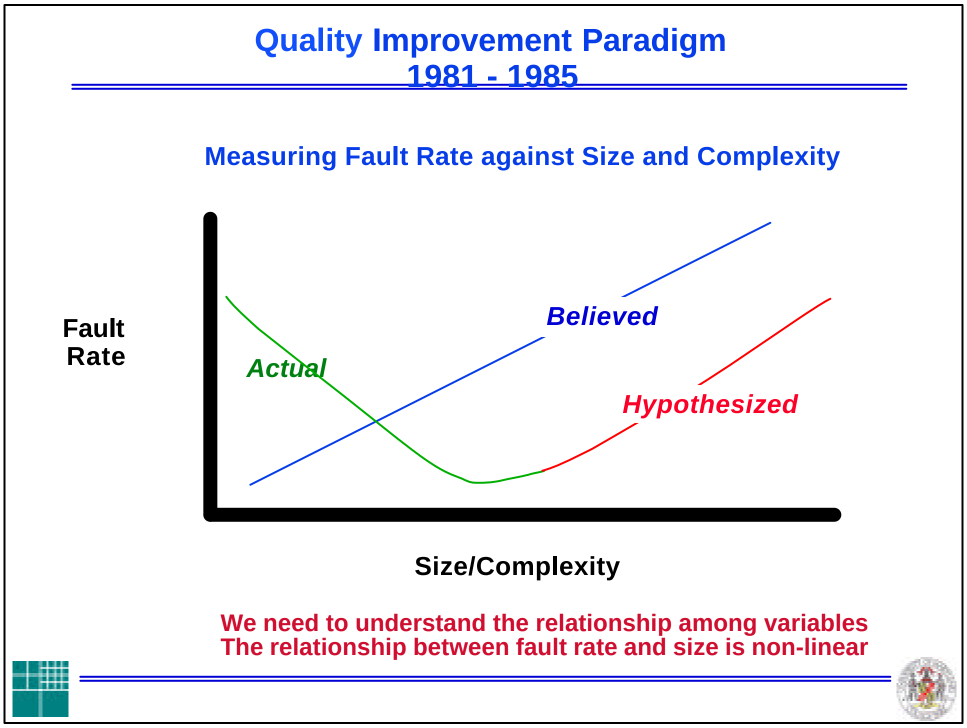
Reusing experience in the form of processes, products, and other forms of knowledge is essential for improvement

Can drown in **too much data,** especially if you don't have goals

...

Developed the QIP as:

Characterize, Set goals, Choose process, Execute, Analyze, and **Record**



Quality Improvement Paradigm 1981 - 1985

Measuring Fault Rate against Size and Complexity

We need to understand the relationship among variables
The relationship between fault rate and size is non-linear

Quality Improvement Paradigm
1986 - 1990

Characterize/Understand

Capturing experience in models

Set Goals

Goals and Models commonplace driver of measurement

Built SME, a model-based experience base with dozens of projects

Select Process

Tailored and evolved technologies based on experience

Experimentation and feedback made explicit in the QIP

Execute Process

Embedded data collection into the processes

Analyze

Demonstrated various (process, product) relationships

Package

Developed focused tailored packages, e.g., generic code components

Learned to transfer technology better through organizational structure, experimentation, and evolutionary culture change

Quality Improvement Paradigm
1986 - 1990

Learned

Experience needs to be **evaluated, tailored**, and **packaged** for reuse

There is a **tradeoff between reuse and improvement**

Software processes must be put in place to support the reuse of experience

A variety of experiences can be reused, e.g., process, product, resource, defect and quality models

Experiences can be packaged in a variety of ways, e.g., equations, histograms, parameterized process definitions

Packaged experiences need to be integrated

...

Reformulated QIP as:

Characterize, Set goals, Choose process, Execute, Analyze, and **Package**

Evolved GQM to include templates and models

Formalized the organization via the Experience Factory Organization

Quality Improvement Paradigm
1986 - 1990

Evaluating and Integrating Reading

Testing vs. Reading experiment
Reading more effective and efficient than testing

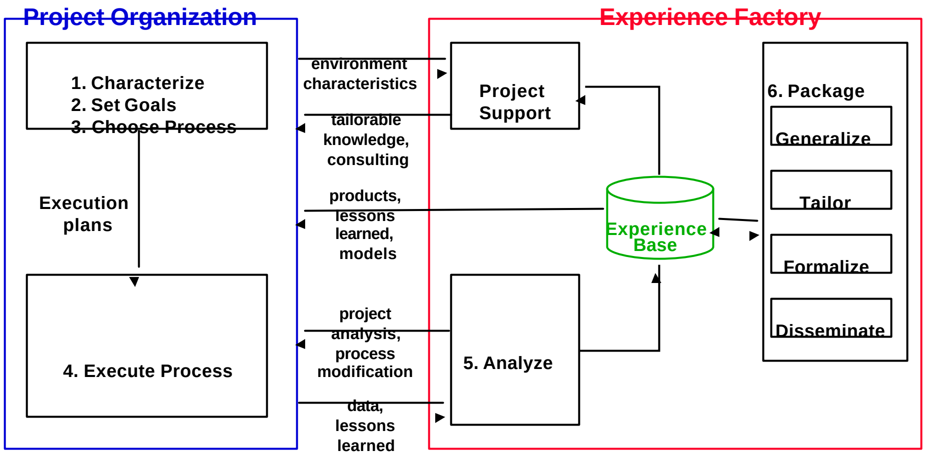
Reading in Practice
Reading had little effect

Reading as part of Cleanroom at the University
Reading had a high impact

Reading as part of Cleanroom in the SEL
Reading had a high impact

How a technology is packaged and integrated has a strong effect
Reading more effective when not followed by testing

THE EXPERIENCE FACTORY ORGANIZATION



Project Organization

1. Characterize
2. Set Goals
3. Choose Process

Execution plans

4. Execute Process

environment characteristics

tailorable knowledge, consulting

products, lessons learned, models

project analysis, process modification

data, lessons learned

Experience Factory

Project Support

5. Analyze

Experience Base

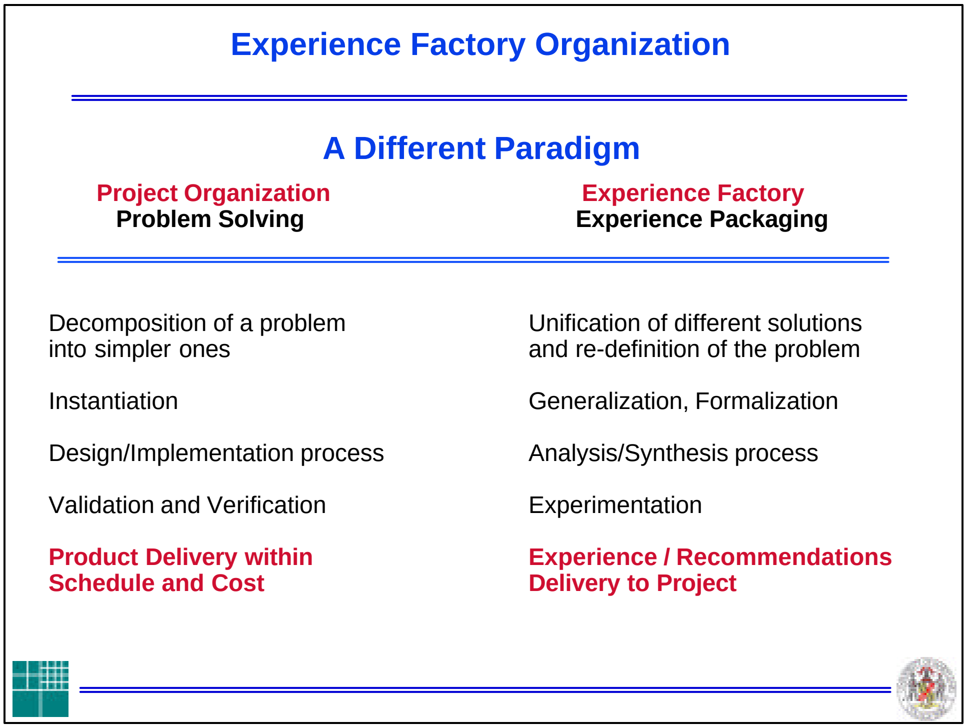
6. Package

Generalize

Tailor

Formalize

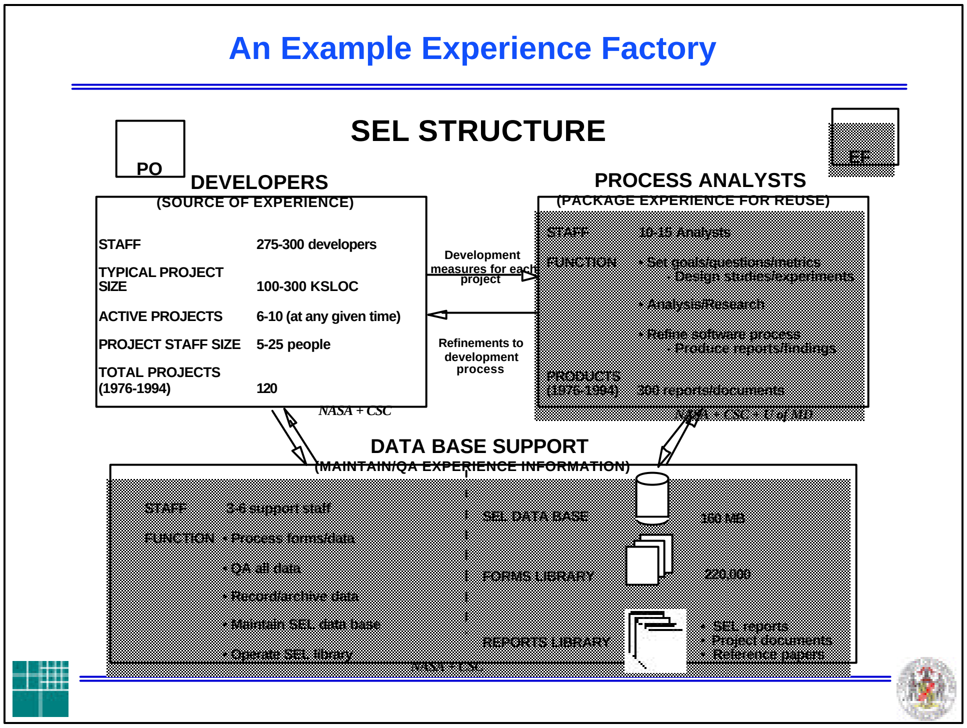
Disseminate



Experience Factory Organization

A Different Paradigm

Project Organization **Problem Solving**	**Experience Factory** **Experience Packaging**
Decomposition of a problem into simpler ones	Unification of different solutions and re-definition of the problem
Instantiation	Generalization, Formalization
Design/Implementation process	Analysis/Synthesis process
Validation and Verification	Experimentation
Product Delivery within Schedule and Cost	**Experience / Recommendations Delivery to Project**



An Example Experience Factory

SEL STRUCTURE

PO

DEVELOPERS
(SOURCE OF EXPERIENCE)

STAFF	275-300 developers
TYPICAL PROJECT SIZE	100-300 KSLOC
ACTIVE PROJECTS	6-10 (at any given time)
PROJECT STAFF SIZE	5-25 people
TOTAL PROJECTS (1976-1994)	120

NASA + CSC

Development measures for each project

Refinements to development process

EF

PROCESS ANALYSTS
(PACKAGE EXPERIENCE FOR REUSE)

STAFF	10-15 Analysts
FUNCTION	• Set goals/questions/metrics - Design studies/experiments • Analysis/Research • Refine software process - Produce reports/findings
PRODUCTS (1976-1994)	300 reports/documents

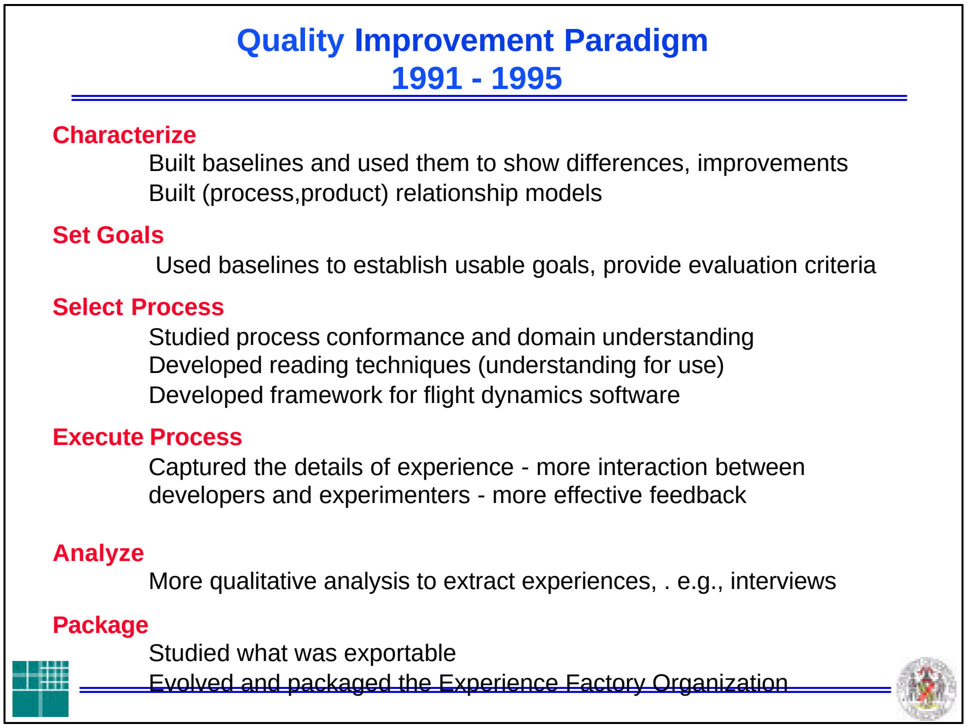
NASA + CSC + U of MD

DATA BASE SUPPORT
(MAINTAIN/QA EXPERIENCE INFORMATION)

STAFF	3-6 support staff
FUNCTION	• Process forms/data • QA all data • Record/archive data • Maintain SEL data base • Operate SEL library

SEL DATA BASE	160 MB
FORMS LIBRARY	220,000
REPORTS LIBRARY	• SEL reports • Project documents • Reference papers

NASA + CSC



Quality Improvement Paradigm
1991 - 1995

Characterize

Built baselines and used them to show differences, improvements
Built (process,product) relationship models

Set Goals

Used baselines to establish usable goals, provide evaluation criteria

Select Process

Studied process conformance and domain understanding
Developed reading techniques (understanding for use)
Developed framework for flight dynamics software

Execute Process

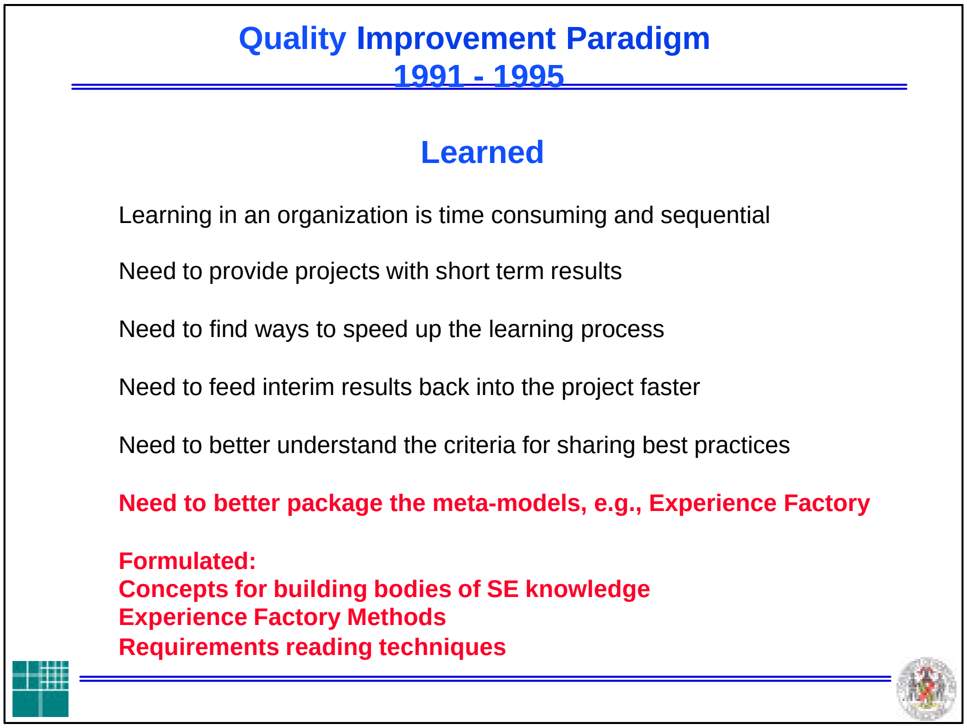
Captured the details of experience - more interaction between developers and experimenters - more effective feedback

Analyze

More qualitative analysis to extract experiences, . e.g., interviews

Package

Studied what was exportable
Evolved and packaged the Experience Factory Organization



Quality Improvement Paradigm
1991 - 1995

Learned

Learning in an organization is time consuming and sequential

Need to provide projects with short term results

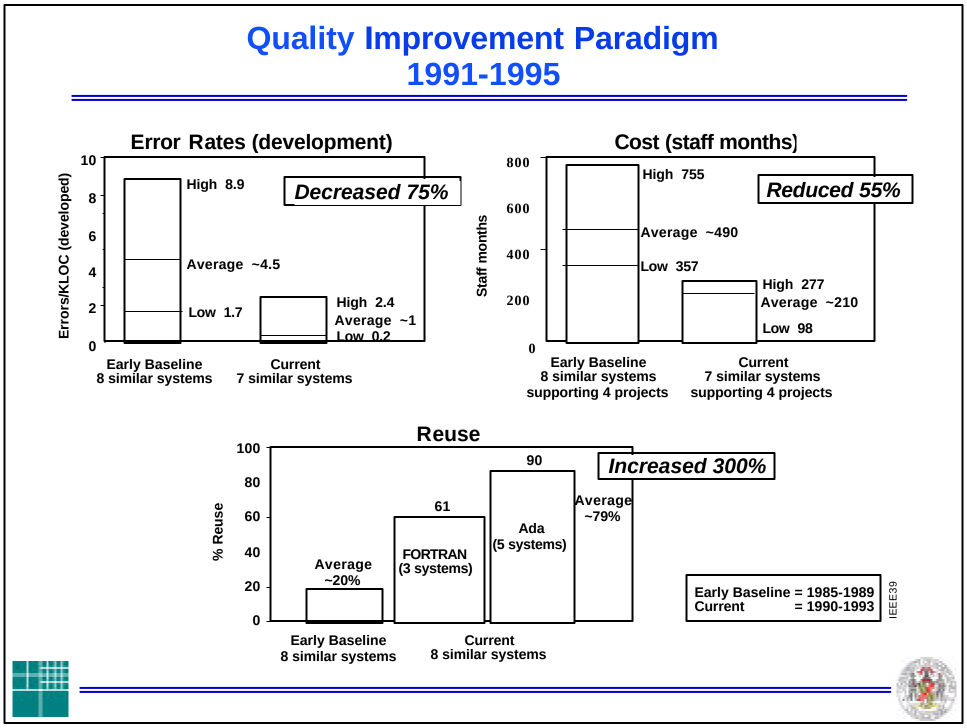
Need to find ways to speed up the learning process

Need to feed interim results back into the project faster

Need to better understand the criteria for sharing best practices

Need to better package the meta-models, e.g., Experience Factory

Formulated:
Concepts for building bodies of SE knowledge
Experience Factory Methods
Requirements reading techniques



Quality Improvement Paradigm 1991-1995

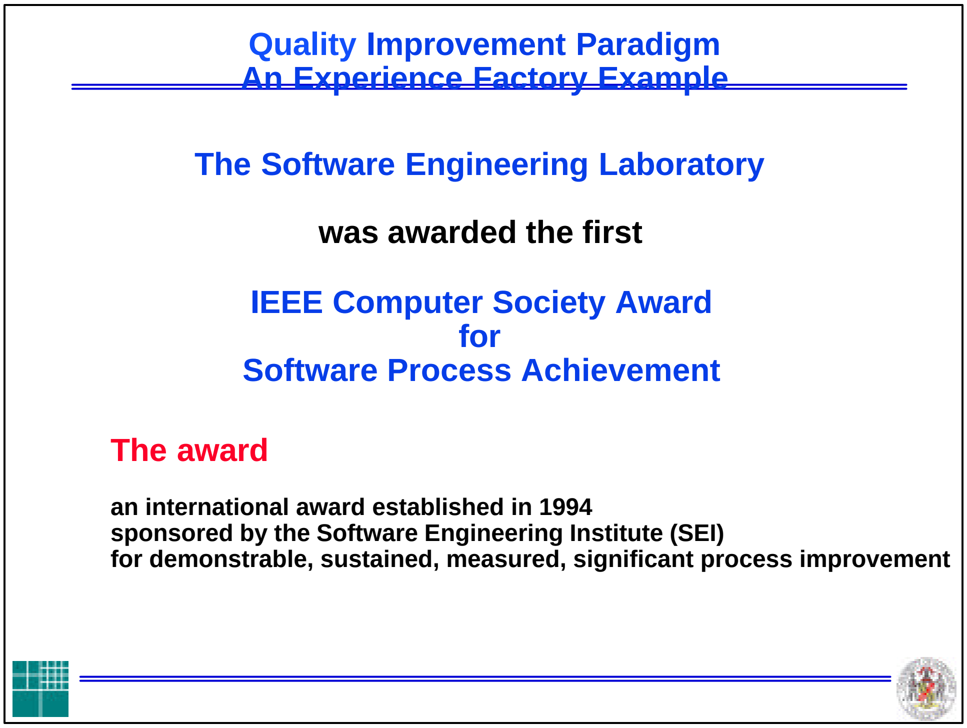
Error Rates (development)

Cost (staff months)

Reuse

Early Baseline = 1985-1989
Current = 1990-1993

IEEE39



Quality Improvement Paradigm
An Experience Factory Example

The Software Engineering Laboratory

was awarded the first

IEEE Computer Society Award
for
Software Process Achievement

The award

an international award established in 1994
sponsored by the Software Engineering Institute (SEI)
for demonstrable, sustained, measured, significant process improvement

Effects of the SEL Activities
Since 1996

Continuous Improvement in the SEL

Decreased **Development Defect rates** by
75% (87 - 91) **37%**(91 - 95)
Reduced **Cost** by
55% (87 - 91) **42%** (91 - 95)
Improved **Reuse** by
300% (87 - 91) **8%** (91 - 95)
Increased **Functionality** five-fold (76 - 92)

CSC

officially assessed as CMM level 5 and ISO certified (1998),
starting with SEL organizational elements and activities

Fraunhofer Center

for Experimental Software Engineering
was created in Maryland in 1998

SEL Studies
Information Systems Center at NASA

Since 1996

ISC Baseline and Measurement

characterize processes products and people

effort and defect prediction models for the various branches

core metrics for contracting and development

COTS Studies

study and evolve the SEL COTS process

define classification schemes for COTS integration

build cost estimation models for COTS development

Reuse/Frameworks

defining a framework-based product line for flight software

Reading Techniques

perspective-based requirements reading

object oriented design reading

SEL Studies
Information Systems Center at NASA

Since 1996

Process/Product Improvement

integration of PSP into the EF concept

study of the effects of EF on achieving higher levels of CMM

Domain Analysis and Technology Transfer

methods for combining results from one organization to another

methods for knowing how to share and tailor best practices

Experience Factory Techniques

methods for packaging experiences and building an experience base

structured interview techniques

combining qualitative and quantitative analysis techniques

study process conformance and domain understanding

Expanding the Learning Organization
The Fraunhofer Center since 1998

Expanding the SEL concepts to other organizations

- Working with small, mid-size companies to improve software business
  - Maryland Software Industrial Consortium
  - Small Business Learning Organization/CMM
- Building Experience Factories
  - Software Experience Center
  - Experience Factory Support
  - Experience Packaging Support
  - Teaming for third party support
- Understanding and generating models
  - IV&V and ROI
  - Software architecture and requirements change
  - Experience Management System

Maturing the Improvement Paradigm

Conclusion

Since 1976 we have **learned a great deal** about software improvement

Our learning process has been **continuous and evolutionary** like the evolution of the software development process itself

We have **packaged** what we have learned into our process, product and organizational structure

The evolution is supported by the **symbiotic relationship between research and practice**

It is a relationship that requires **patience and understanding** on both sides, but when nurtured, really pays dividends!

Maturing the Improvement Paradigm

Conclusion

Improvement of software competence is an essential business need

We need to

- build **software core competencies** as part of our overall **business strategy**
- create **organizations for continuous learning** to improve software competence
- generate a tangible **corporate asset:** an **experience base of competencies**

QIP/GQM/EF represents a promising approach

- a **Lean Software Development** concept
- compatible with **TQM** concepts
- offering a **level 5 organizational structure**