Process Improvement In Retrospective

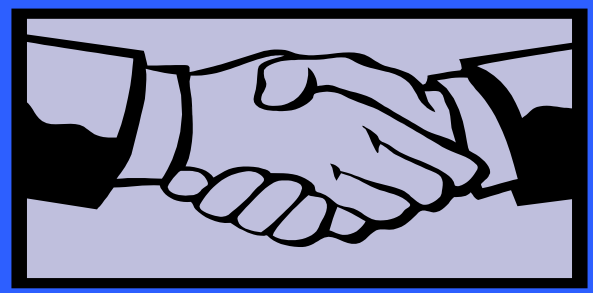
(Lessons Learned from Software Projects)

SEPG Conference March 2005, Seattle, Washington

Developed By John D. Vu
Technical Fellow & Chief Engineer
The Boeing Company

Purpose

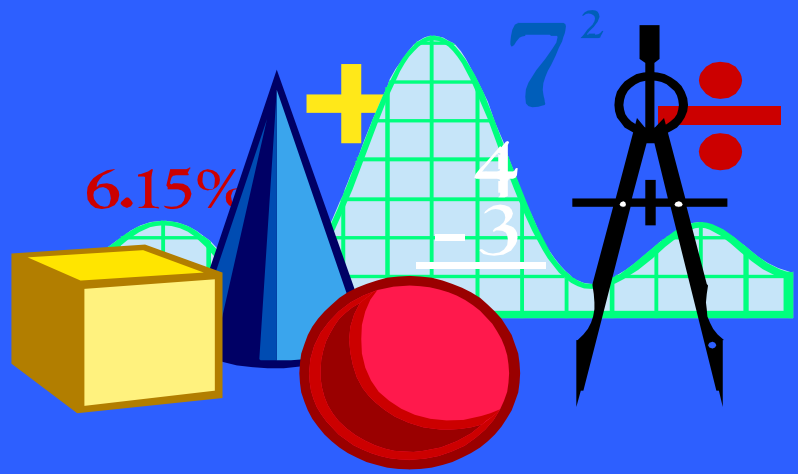
The purpose of this presentation is to share lessons learned from software projects and process improvement activities



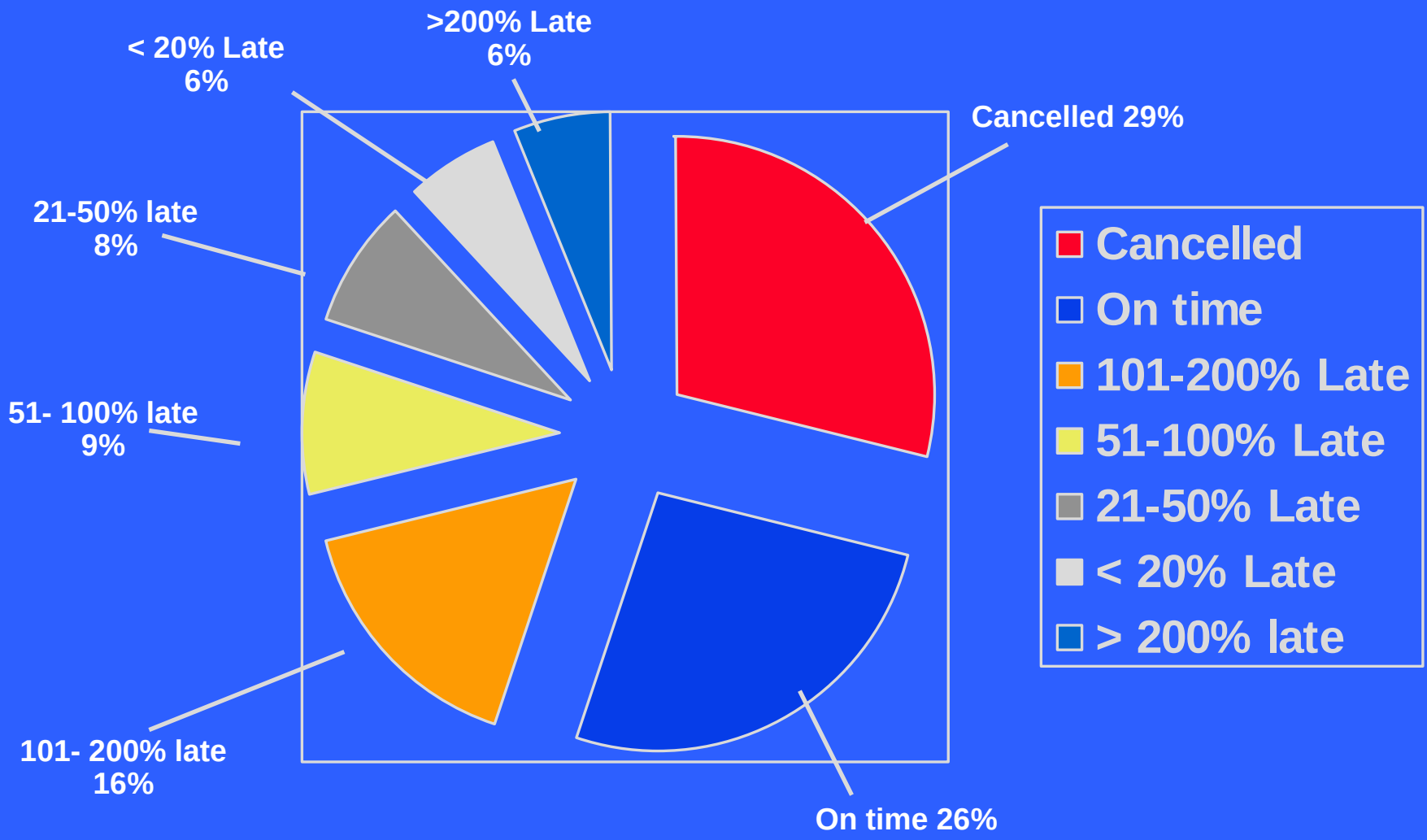
Agenda

- State Of The Practice
  - Industry Data
  - Process Improvement
- Lessons Learned
  - Software Development
  - Software Process Improvement
- Questions & Answers

Why Do You Need Software Process Improvement ?



Disappointing Software Project Outcomes



* Software Industry Benchmarking Study 2001

Software Project Inefficiencies

Manager and customer expectations are often unrealistic and unachievable

Most Project Managers do not receive adequate training

Average software developer reads less than 1 professional book per year and subscribes to no professional journals

Schedule is still the most important factor at the expense of quality, cost and efficiency

Most project estimates are based on expectations rather than historical data

*** Software Industry Benchmarking Study 2001**

Current Practices

“I'd rather have it wrong than late. We can always fix it later”

A software project manager

“Why software costs too much and takes too long?”

A Chief Financial Officer

“The bottom line is the schedule. My promotions and raises are based on meeting schedule.”

A Program Manager

*** Software Industry Benchmarking Study 2001**

State Of Software Projects In The U.S.

Much of the $250 billion in annual U.S. software development spending is wasted, late, incomplete, or spent on canceled projects:

53% ($132.5 billion)	**are considered over budget, delayed, and less functional than planned.**
31% ($77.5 billion)	**are considered impaired and must be canceled.**
16% ($40 billion)	**are completed within budget, on time, and with all functions included.**

Source: The Stanton Group (1999)

State Of The Software Practice Today

- Software cost and schedule are not accurately estimated
- Software is often shipped full of defects
- Software development work is high stress and requires long hours
- Software development incurs large expenses due to rework and testing
- Accurate project status is difficult to get

* Software Industry Benchmarking Study 2001

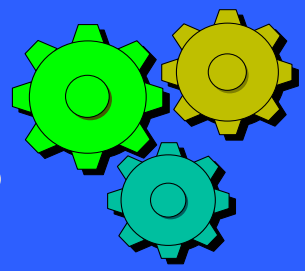
BOEING®

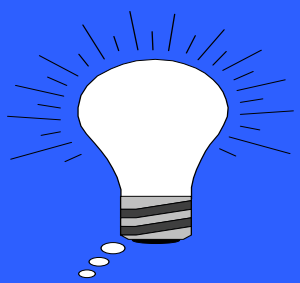
Software Improvement

Improving Software Practices is Business ' Most important challenge

Why Improve Process?

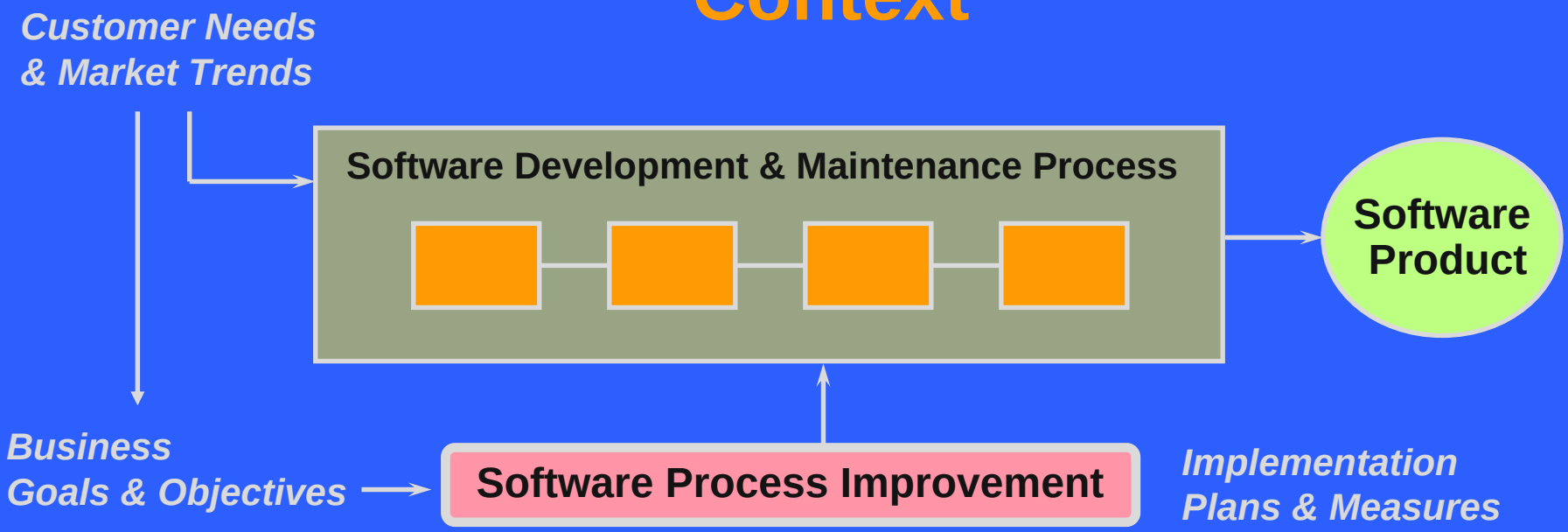
The quality of a software product is governed by the quality of the processes used to develop and maintain it.





To improve the quality of the product, one must improve the quality of the processes used to create the product.

Context



Software Development & Maintenance Process:

The development & maintenance of products that meet the needs of customers and markets

Software Process Improvement:

Application of technology and disciplines to improve software development and maintenance processes

The SW-CMM

The Software CMM has become the de-facto standard for assessing and improving software processes

SW-CMM (earlier versions)	– 1988
SW-CMM V1.1	– 1991
CMMI V1.0	– 1998
CMMI V 1.1	– 2001

Process Improvement Using CMMs is > 15 years

Failure To Improve

Industry data found that

- **72% of organizations report little or no success in software improvement after an assessment**
- **83% of organizations abandon their improvement efforts in the first 3 years**
- **57% of organizations that abandon improvement efforts restart them in the future**
- **Less than 1% of organizations claiming success in process improvement report improvement data**

*** Software Industry Benchmarking Study 2001**

Why Do So Many Improvement Efforts Fail ?

1. Over-emphasis on having assessments but not much focus on the commitment to make improvement happen

2. Focus mostly on maturity levels without clear direction and measurable objectives

3. Lack of a skilled infrastructure to coordinate and manage improvement activities

4. Confusion between buzzword terminology and actual practices

5. Implementation of improvement solutions is poorly managed

* Software Industry Benchmarking Study 2001

Retrospective

Some People Still Believe That...

"Let's conduct an assessment, then senior management will commit to process improvement".

Build it and they will come ....

Many organizations hurry into having an assessment before obtaining commitment from senior management

Many organizations want to start quickly and not take the time to work out a detailed plan for process improvement

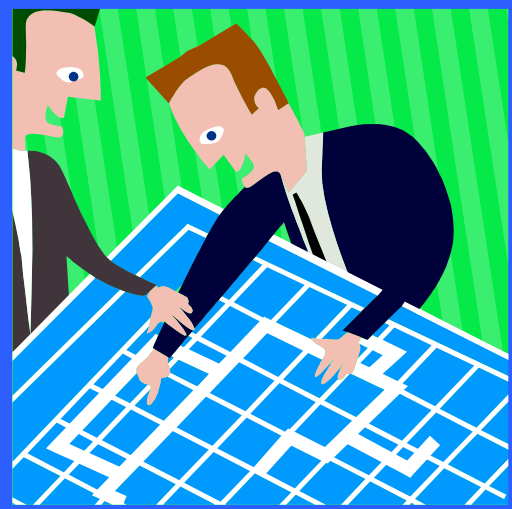
Assessment Field of Dreams…

However, The Fact Is...

Senior Management's commitment is the most essential element in the success or failure of software process improvement

Organizations must obtain commitment from senior management before starting any improvement activities

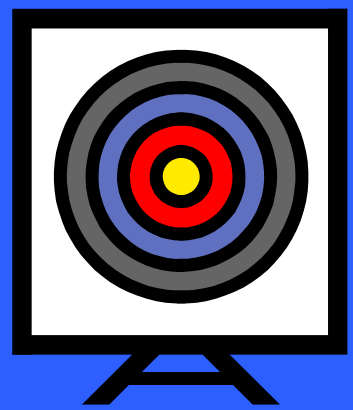
Organizations must take the time to work out a detailed plan with clearly defined goals and objectives to avoid false starts, unsustainable progress and unclear expectations of the results



Commitment vs. Direction

Direction:

Go ahead, do it



Commitment:

Managers: Understand ➡ Accept ➡ Support

Practitioners: Aware ➡ Participate ➡ Action

Pre-Improvement Check List

- ☐ Do you have management commitment for process improvement?
- ☐ Do you have business reasons for undertaking process improvement?
- ☐ Have you identified sponsors, process owners and change agents?
- ☐ Have you established an infrastructure to manage the improvement effort?
- ☐ Have you communicated the context for improvement with the people in the organization?
- ☐ Have you started to build support for the process improvement effort?

Some People Still Believe that...

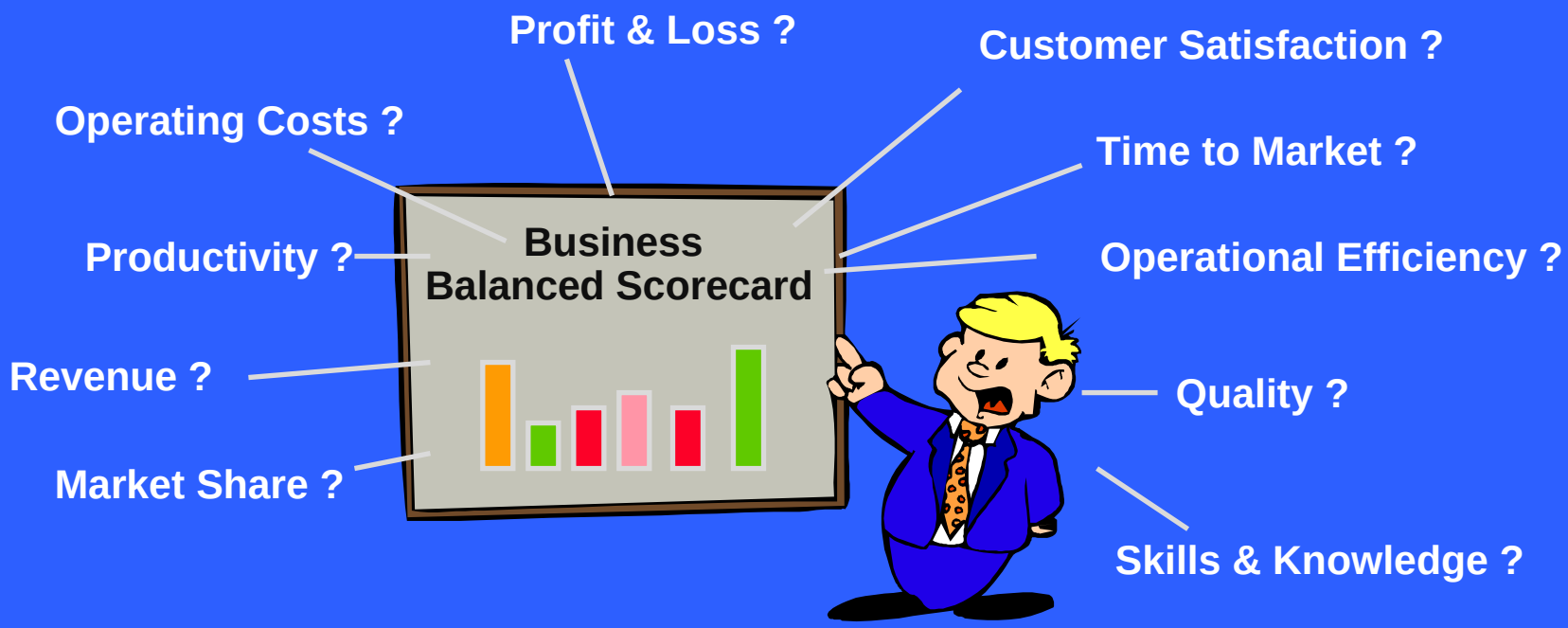
"The goal of process improvement is the maturity level"

Many organizations confuse maturity levels with improvement objectives

Some organizations believe the maturity level is the "miracle" that can make improvement happen

However, The Fact Is...

Maturity levels are meaningless if they cannot be explained in terms of business objectives



Maturity levels are milestones on the improvement journey and are never intended to be the goal

The Need To Link Improvement To Business Value

Management needs to understand that the goal of improvement must be based on current business needs and NOT on maturity levels.

Management must communicate process improvement in terms of…

Business Values:

- **Increase Quality**
- **Reduce Cost**
- **Reduce Cycle Time**
- **Increase Productivity**
- **Increase Customer Satisfaction**
- **Increase Employee Satisfaction**

If process improvement is not directly aligned with business goals and objectives, it has no value to management and should not be allowed to continue.

Some People Still Believe that...

"The main activity of the Software Engineering Process Group (SEPG) is to conduct assessments"

Many Software Engineering Process Groups over-emphasize conducting assessments and do not focus enough effort on making improvement happen

"Assessments are Us" symptom

However, The Fact Is...

The Software Engineering Process Group (SEPG) is a group of highly skilled change agents who facilitate and coordinate all improvement activities of an organization

Maintain Organization Standard Software Processes

Coordinate Process Improvement Activities

Evaluate & Transfer New Technology

Collect & Analyze Measurement Data

Conduct Appraisals & Develop Improvement Plans

Improvement Effectiveness

Improvement is a serious business that requires an effective infrastructure that:

- **Understands the organization's business objectives**
- **Has a plan for its work**
- **Has a tracking system in place to track progress**
- **Collects project and process data**
- **Measures the quality and productivity of its own work**
- **Knows the organization's training needs**

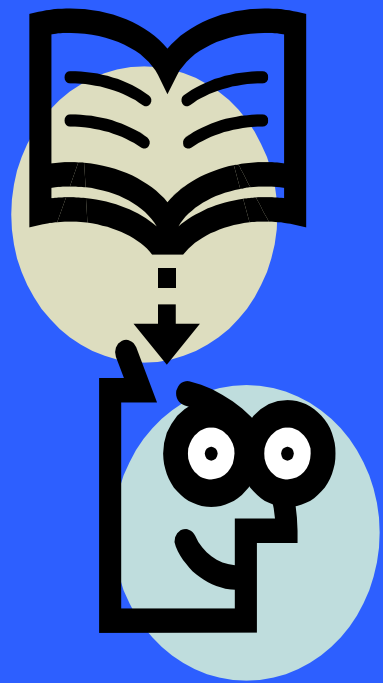
Some People Still Believe that...

"Highly skilled people are not needed to do improvement work. Any available people can do process improvement"

Many organizations select people who may not have the right skills and motivation for leading process improvement.

Without actual engineering and improvement experience, many rely solely on book knowledge, and will focus on writing plans without spending time solving the organization's critical issues.

Improvement can become a documentation exercise.



However, The Fact Is...

Process Improvement requires the most experienced and skilled people in the organization.

SEPG members need to be communicators and motivators; they must have the skills and experience to do their job successfully.

SEPG members must be selected from projects and rotate in and out of process improvement activities for experience.

SEPG is a challenging job that requires "Change Agent" skills and should never be treated as a dead-end job.

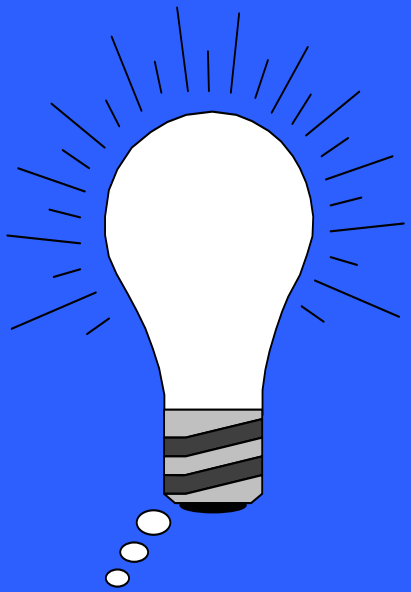
Where Do SEPG Members Come From ?

SEPG members should come from projects and have experience throughout the software life cycle and the business processes of the organization. They should have experience with multiple frames of reference and their backgrounds should complement each other.

Membership:

1. Full time members (Rotate every 2 -3 years)
2. Part time members (Few hours per week)
3. Special assignment members (Full time for a few months)

Question



How Many SEPG Members Does It Take To Change A Light Bulb?

Answer

Only One,
But
The Light Bulb Has To Be
Willing To Change !

Some People Still Believe that...

"Just document every CMMI practice and select a few small projects to represent the organization, and then we can pass the appraisal and claim that the entire organization has achieved a high level of maturity"

This "improvement by documentation in a small sample of projects approach" does not solve critical organization issues.

The organization is only interested in a maturity level and is not serious about real improvement. This "business as usual" attitude will not fool anybody. Technical people know how to read it, and Lead Assessors are trained to catch it.

CMMI is NOT a cook book or a requirements specification.

However, The Fact Is...

The appraisal process is more than reviewing documentation. Processes must be practiced, used, measured and continuously improved to achieve the organization's business goals and objectives.

Process improvement is about solving critical problems and not about achieving meaningless maturity levels.

SCAMPI appraisals focus on both process compliance and organizational performance.

The SCAMPI method requires organizations to disclose the scope of the appraisal and, improvement results must be measured as part of the business value stream.

Some People Still Believe that...

"There is no need to improve the process. Just throw a few "good people" at the problem and it can be solved"

There is a notion that a few "good people" can do a far better job than the rest.

These "super human beings" know how to solve problems intuitively; just program thousands of lines of code per day and the organization will do just fine.

Where can you find these few "good people"?

The organization is looking for a few "good people"

However, The Fact Is...

Being able to staff your project with the best people is good, but to support them with a stable environment and effective management systems is far better.

Project management requires skill, experience and effective processes to make things happen

Most projects fails because of wishful thinking, unrealistic schedules and forever-changing requirements

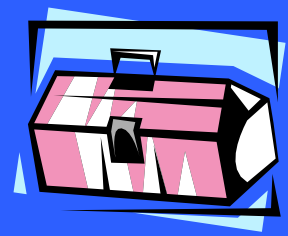
People need a stable environment and effective processes to be productive

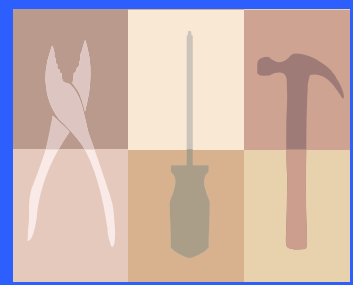
Some People Still Believe that...

“Organizations can improve faster and achieve higher maturity levels by buying more tools”

Some organizations believe that they can improve by having more tools.

Some tool vendors claim their tools can get an organization to a higher level of maturity.





However, The Fact Is...

Automation of poorly defined processes will produce poorly defined results much faster.

It takes time, skill, and resources to improve an organization's processes.

Unplanned process automation is wishful thinking.

Process improvement should be made in incremental steps and should NOT be automated until it is fully institutionalized.

Some People Still Believe that...

"Just document the process -- improvement will come! The CMMI requires documentation, because the process is the document"

This approach results in massive efforts to write standards and procedures instead of solving critical issues.

Are software practitioners suppose to wait patiently for these documents to be completed before they can improve their processes?

Standards and procedures will not lead to process improvement; they have to be used, measured and institutionalized.

However, The Fact Is...

Process improvement is about changing the way people develop software and systems, by practicing a disciplined engineering approach to meeting the organization's business goals & objectives.

SEPG must work with the practitioners to identify training needs and support the improvement of the software and systems engineering disciplines.

SEPG must facilitate and coordinate improvement activities with the practitioners and avoid being document writers.

To improve, all procedures must be used, measured, supported and institutionalized.

Some People Still Believe that...

The software requirements are not clear, but we can work out the details later.

Most software requirements are ambiguous and poorly communicated.

Many engineers want to start the project quickly and not take the time to work out the detailed requirements.

Customer

Engineer

However, The Fact Is...

Work closely and cooperatively with the customer to establish and maintain a firm requirements base. Ensure all requirements are expressed in simple unambiguous terms; quantifiable and verifiable.

Some People Still Believe that...

Software test cases and test scripts need not be defined until just before the testing phase.

Because most projects are late due to efforts in defining interfaces & functional requirements, many engineers defer test cases and test scripts to later phases.

However, The Fact Is...

Test cases and test scripts must be developed during the requirement phase to verify the requirements.
Customers must review and agree to the test cases and scripts.
Engineers must focus on getting good verifiable software requirements before starting preliminary design.

If you cannot test it, you do not understand the requirements

Some People Still Believe that...

Software schedules are generated by experienced managers or professionals, therefore they are realistic and achievable.

Software personnel have a tendency to be optimistic.

Lack of historical data can lead to over or under estimation.

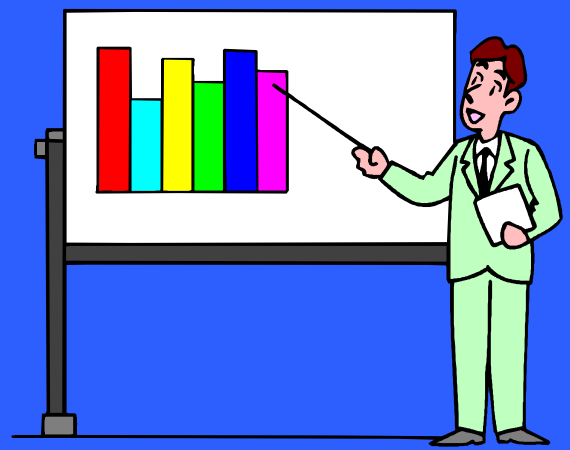
Schedules may be end-point constrained.

However, The Fact Is...

Need to ensure that software schedules have sufficient detail, are independently assessed, and have adequate management commitment.

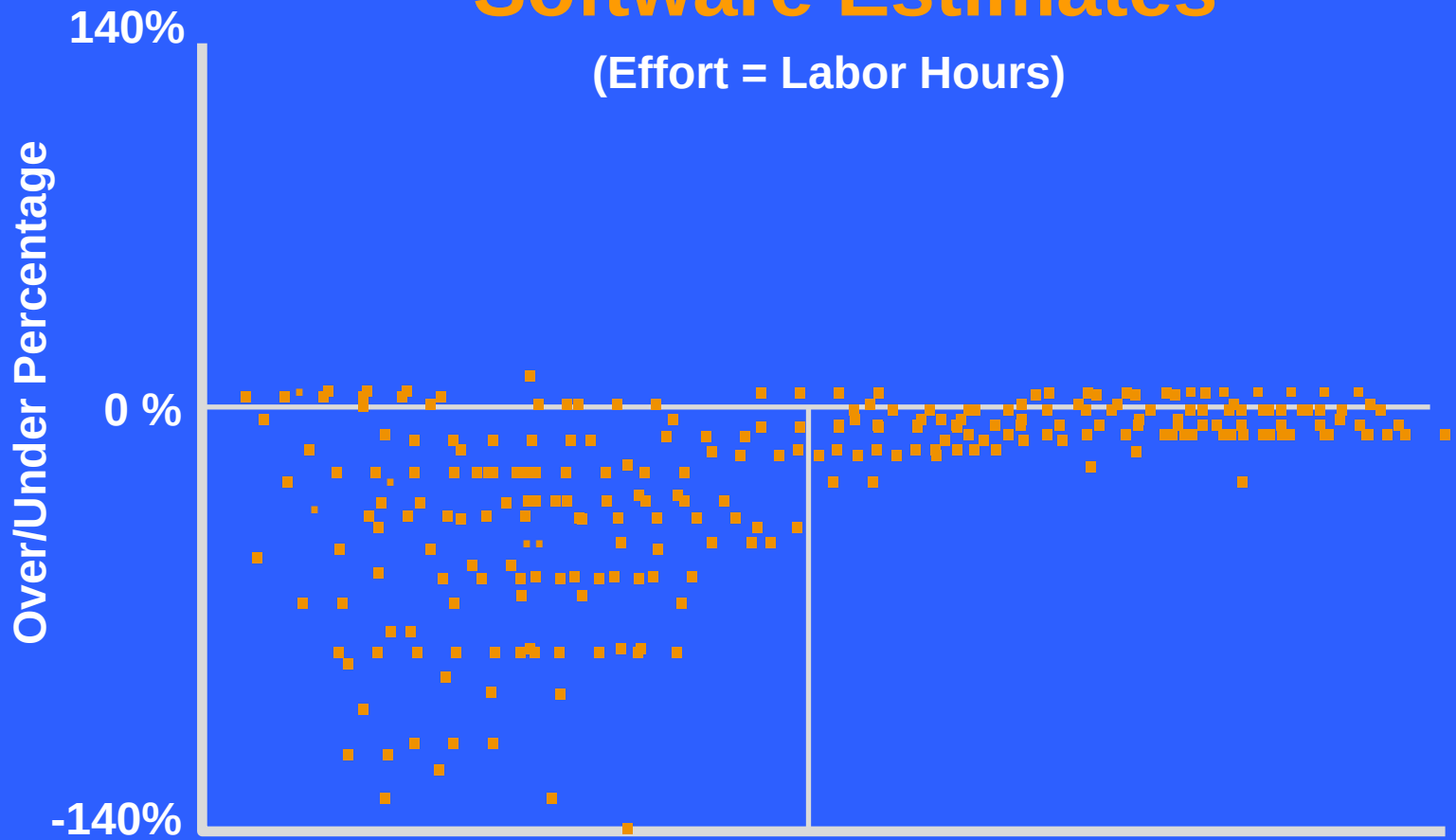
Follow a planning process that achieves a complete definition of all deliverables and tasks and all associated interdependencies.

Maintain a disciplined approach and use historical data.



Software Estimates

(Effort = Labor Hours)



Variance between + 20% to - 145%

Variance between - 20% to + 20%

Based on data collected from 120 projects between 1991 to 2000

Project Planning Checklist

- [ ] Are all requirements documented and meet the exit criteria of requirements management ?
- [ ] Is a project plan established and documented ?
- [ ] Is the project plan based on estimates (size, resources, schedule) ?
- [ ] Are the project estimates based on historical data ?
- [ ] Are software work products (deliverables) identified ?
- [ ] Are software life cycles identified and documented ?
- [ ] Are project risks identified ?
- [ ] Is there a risk mitigation plan ? Is it documented ?
- [ ] Is there a project performance monitoring mechanism in place ?

Some People Still Believe that...

Software project managers were selected for the job and do not need any help.

Many people believe that most software project managers are fully prepared in all areas.

Project managers are too busy to take time to consult with outside personnel.

It is difficult to recognize potential problems when you are busy with cost and schedule pressures.

However, The Fact Is...

An advisory board should periodically review the software development activities.

An advisory board consisting of people with experience from similar projects can help identify potential problems early on.

An effective way to capitalize on Lessons Learned.

The board should meet periodically to review technical progress as well as project schedule and budget status.

Project Management Infrastructure

Organization Management
(Senior Managers)

- Set policy & direction
- Provide resources

Technical Advisory Board
(Technical Leaders, Project Managers)

- Track progress & risks
- Support technical issues
- Advise project team
- Collect lessons learned

Project Managers

- Manage project
- Report progress
- Deliver products

Some People Still Believe that...

There is no need to define user interfaces early on as they do not impact software development.

A poorly designed user interface can constrain the overall system's operational capability and lead to customer dissatisfaction.

Failure to recognize the correlation between a good user interface design and achieving a viable operational concept.

However, The Fact Is...

Need to have agreement with the customer on the user interfaces in parallel with the software requirements and design activities.

Define the user interface requirements at the same time with other functional requirements.

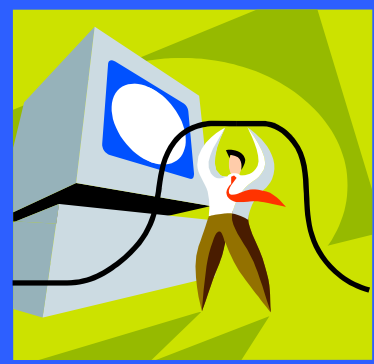
Assure that user interface and operational concepts are developed in parallel and are compatible.

Prototype the critical operational displays and let the user try them out before freezing either the requirements or display formats.

Cost Of Un-usability For End-Users

Many systems are only technically successful and fall short of their expected benefits because the users find them:

- **Difficult to understand and use**
- **Inconsistent**
- **Needing excessive training & retraining**
- **Error-prone**
- **Discouraging to explore**



Costs Of Un-usability For Developers

The typical user interface is about 45% of the current application effort. Wrong assumptions can lead to:

- **False starts**
- **Extensions of the development schedule**
- **Rework - Cost of non-quality**
- **Increases in the complexity of the code**
- **Lower customer satisfaction**

Some People Still Believe that...

We already estimated software size and throughput during the requirements phase and do not need to revisit the data until the code is produced.

Software engineers tend to be optimistic in their estimates.

Failure to continually evaluate and monitor throughput and sizing.

Lack of historical data for project estimates.

Wrong assumptions used in the original estimates.

However, The Fact Is...

Establish a dynamic and realistic estimation model for software size, effort, schedule and throughput, and keep it updated. Use it for project management.

Follow an estimation model and adjust estimates and budget with historical or benchmark data.

Use early prototypes to validate throughput models.

Establish and maintain a risk mitigation plan to accommodate increases in sizing and throughput.

Some People Still Believe that...

The only software developmental activity that needs attention is coding.

Coding is only a small part of the software development activities. It is not an area where projects get into trouble, but it is where many problems (created earlier) start to show up.

Projects never fail because people can not code. Coding is never the root cause.

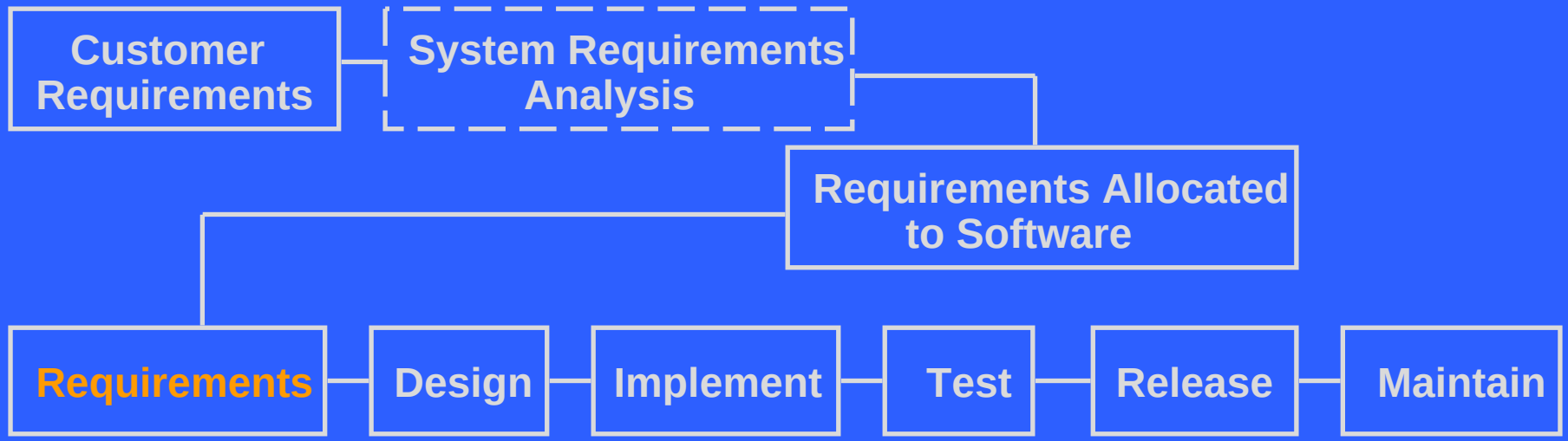
However, The Fact Is...

Organizations must monitor all of the life cycle activities, and use metrics where appropriate.

Monitoring activities includes:
- **Planning**
- **Tracking**
- **Measuring**
- **Validating**
- **Verifying**

Assure the project uses metrics to measure performance, quality, schedule and functionality.

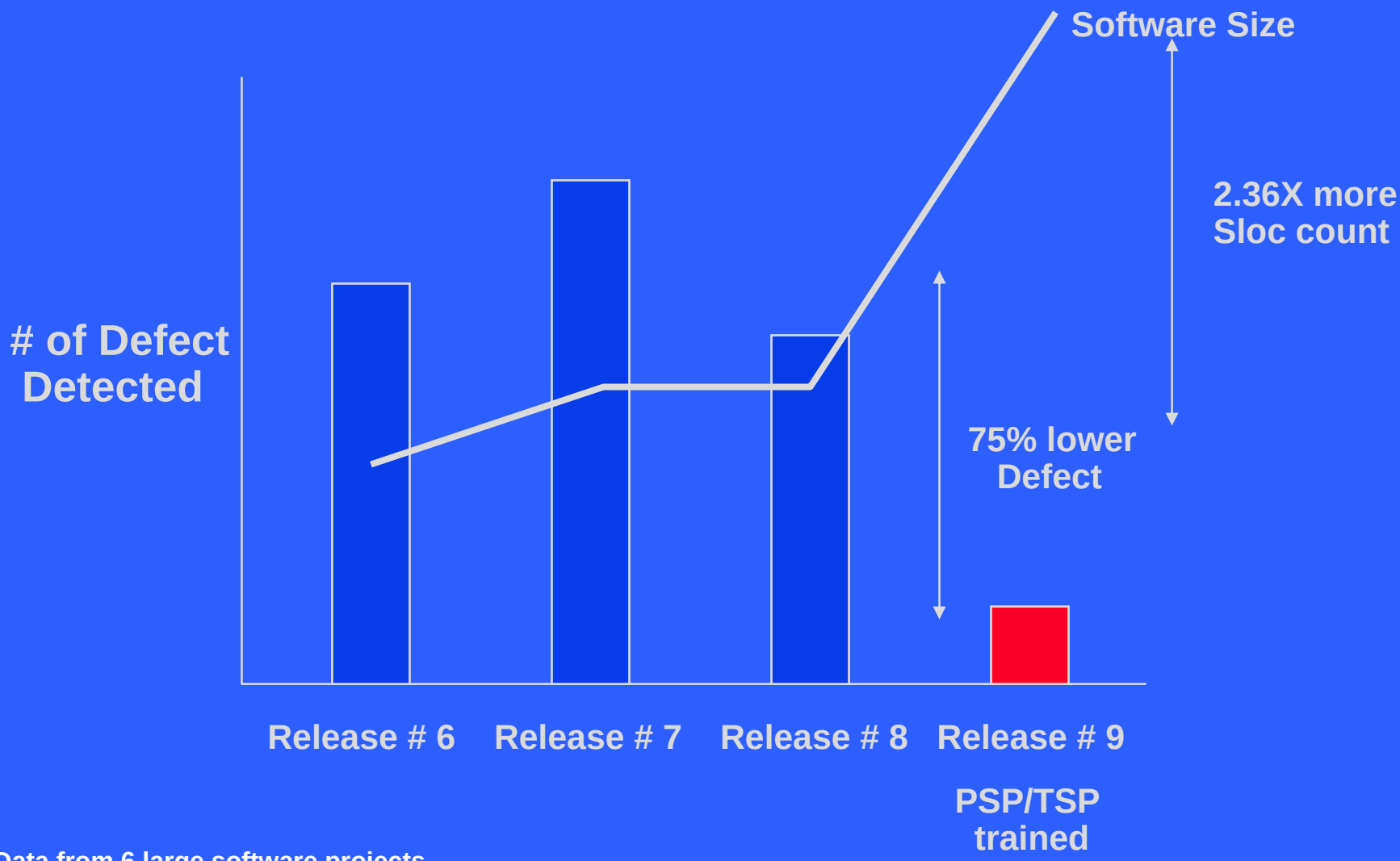
Traditional Software Life Cycle



Effort:	5%	10%	40%	20%	5%	20%
Defects:	45%	25%	15%	10%	1%	4%

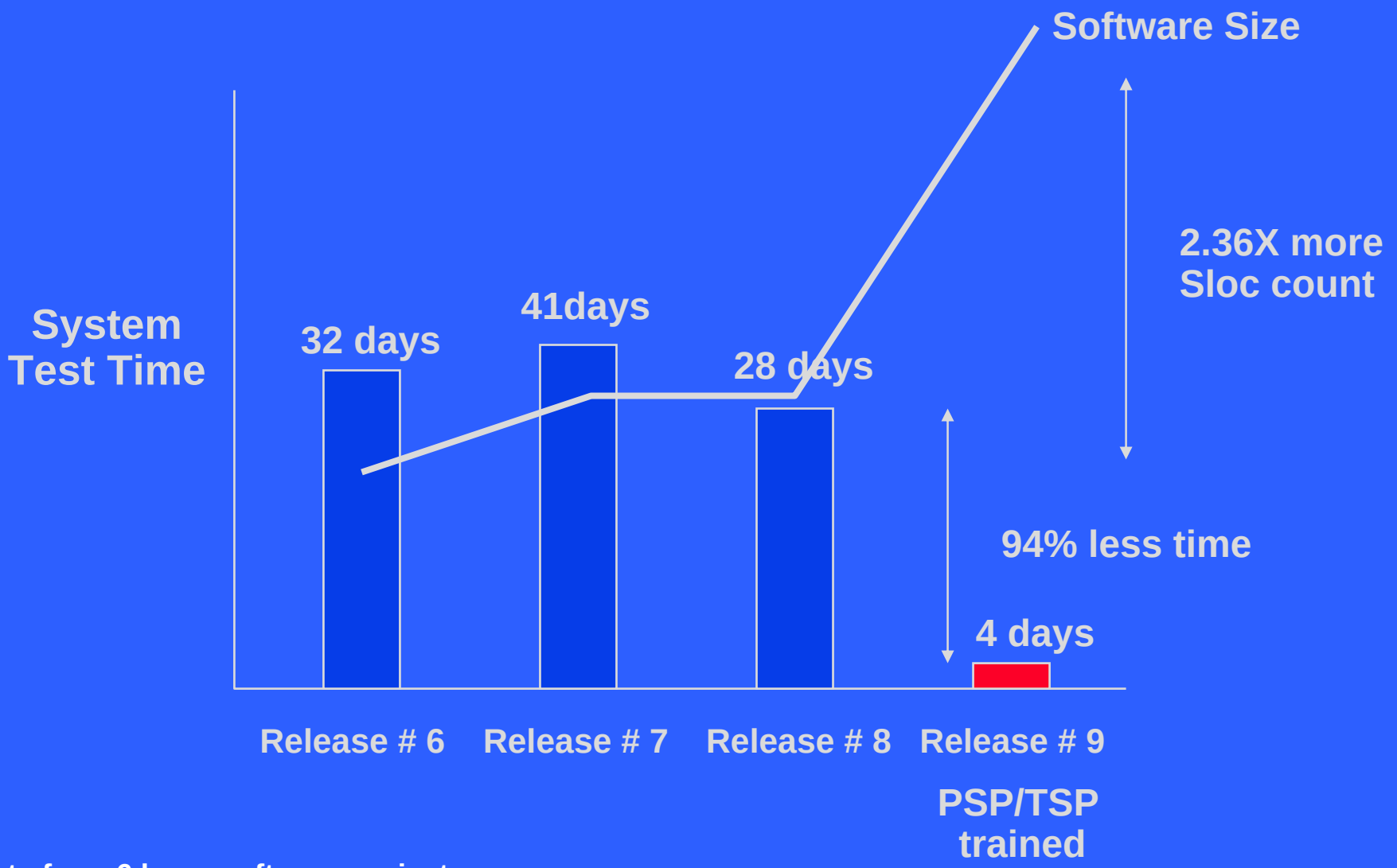
Industry data derived from IEEE software

PSP / TSP Benefits



Data from 6 large software projects

PSP / TSP Benefits



Data from 6 large software projects

Some People Still Believe that...

Software design can proceed prior to the definition of the software external interfaces.

Initiating design without clear definition and control over all of the external interfaces will create significant risk of rework.

Without all of the interfaces clearly identified, allocation of system requirements to software will be inadequate.

Without all of the external interfaces clearly defined it is impossible to evaluate system I/O bandwidth.

However, The Fact Is...

Define and control external interfaces, and obtain agreement from all affected parties prior to software design.

Projects must establish an interface control team to define, document and manage interfaces.

Projects must ensure that interface definition is following a top down approach, i.e. from system level to software component level, prior to software architecture design.

Projects must ensure contractual documentation between customers and suppliers, and include interface definitions for future integration.

Some People Still Believe that...

An organizational standard software process is not needed for our "unique" software project.

Many projects incur unnecessary cost re-inventing the software process.

Many engineers do not want to comply with established standards for "creativity" reasons.

Many engineers do not understand the benefits of following standard processes.

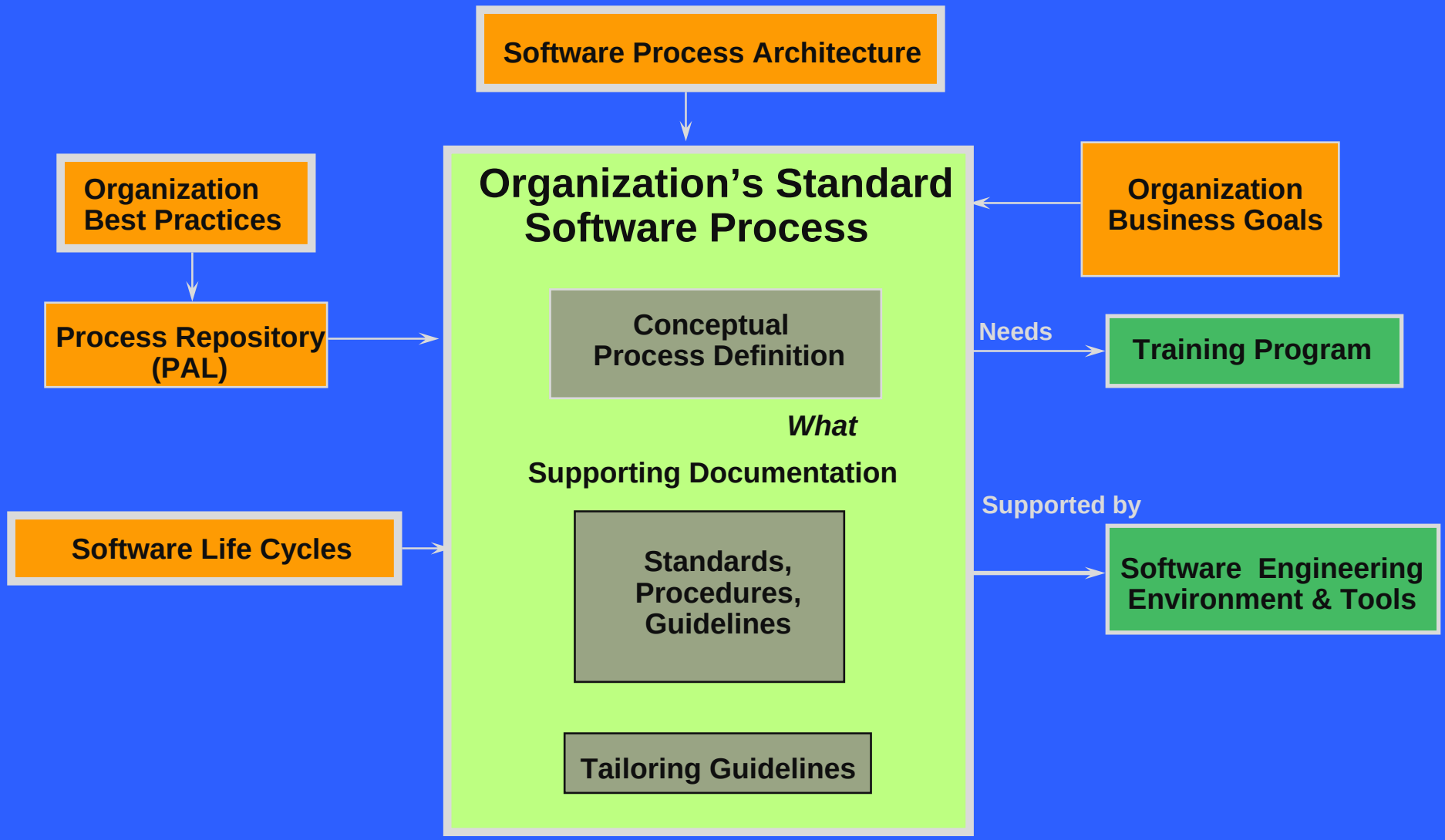
However, The Fact Is...

Organizational standard software processes are established to ensure project consistency, improve product quality, avoid unnecessary project costs and reduce risks.

Established organizational standard processes do not prevent "creativity" since engineers can tailor them to fit the project as appropriate.

Identify project "unique standards" as additional to (not replacement for) the organization standard software processes.

Organizational Standard Process



Question

What Are The Indications Of A Successful Process?

Answer

Does not go away when resources gets scarce or people leave

Gets improved by people who are actually using it

People following it see real benefits, real changes

People apply it to something else

People do it subconsciously (Fully Institutionalized)

There are clear measures of the change (Trends, metrics, ROI data)

All practitioners use it (NOT imposed by management)

People talk about other issues rather than process issues

A Successful Process

Some People Still Believe that...

The software engineer is the best judge of when his or her milestones are complete.

Many projects have experienced unexpected rework as a result of an engineer passing incomplete work downstream.

Task completion is judgmental, rather than quantifiable.

Lack of defined metrics and strong project management oversight.

Task completion becomes a matter of assertion rather than observation and measurement.

However, The Fact Is...

Have quantifiable evidence of completion agreed to by the customer for each task.

Need to establish a planning process that ensures that the completion criteria for all tasks are measurable, documented and agreed to by all affected parties before initiating.

Enforce a tracking system that ensures quantifiable closure criteria exists and is used for management visibility.

Some People Still Believe that...

Reusing software is easy.

Adapting existing software components to new applications is easy; If it does not fit, modify it until it fits.

Any code can be reused since it's "modularized" already.

Many engineers do not fully understand the software reuse concept.

However, The Fact Is...

Prior to deciding to reuse software, you must ensure that it meets the requirements, without modification.

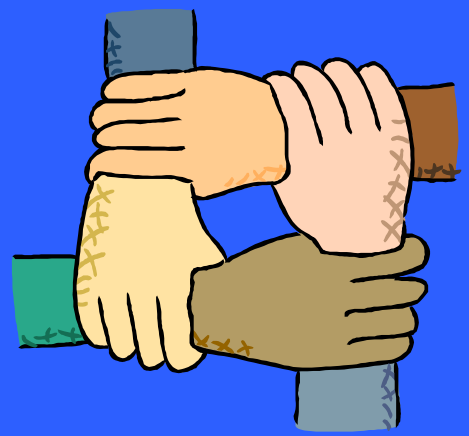
Develop a contingency plan if the reused software does not function as expected.

Opportunistic reuse:

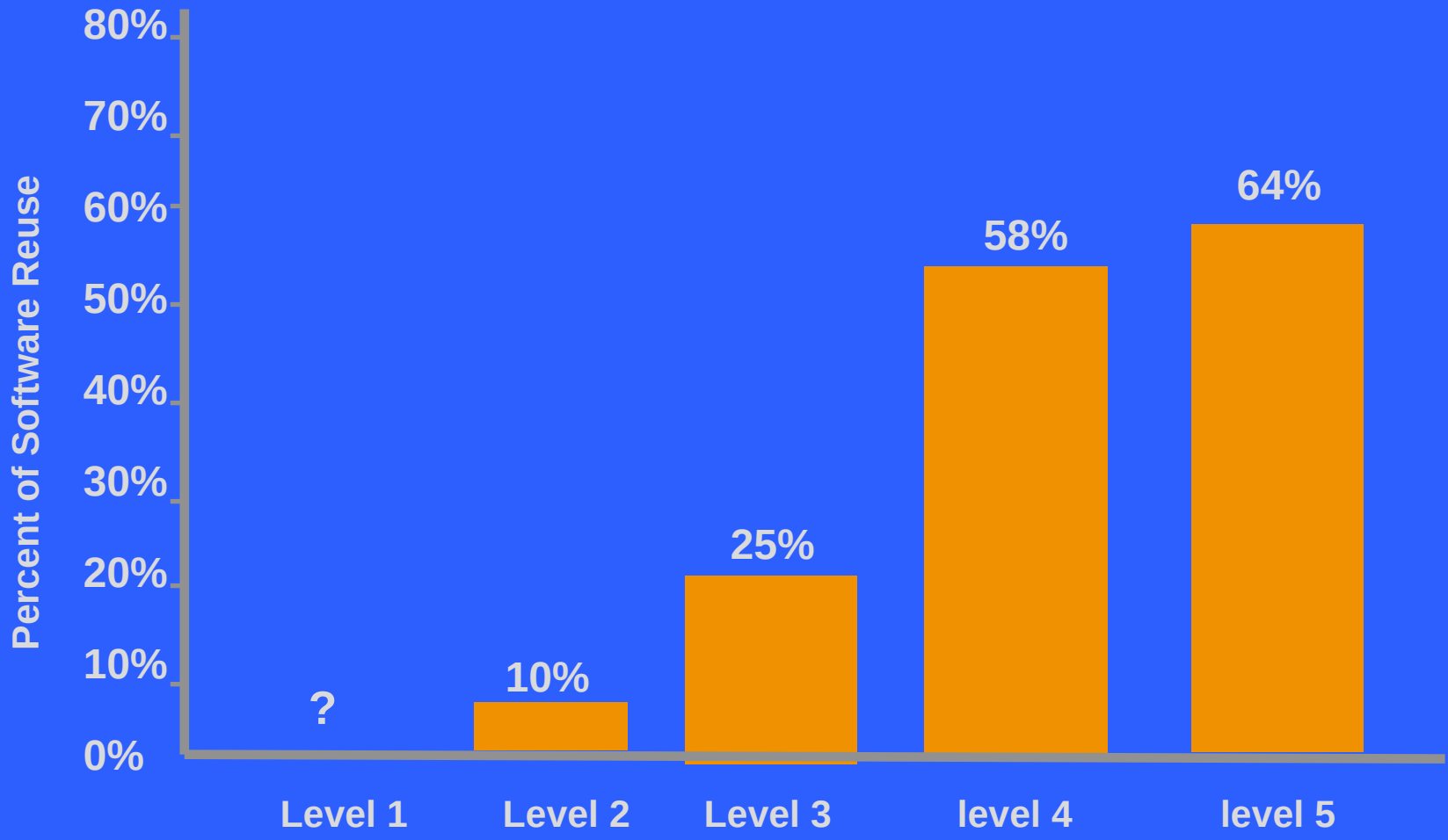
Adapt existing software components to new applications by modifying them. It is not cost effective since you still have to modify and test it.

Systemic reuse:

Reuse “well-defined” and “well designed” software components without any modification.

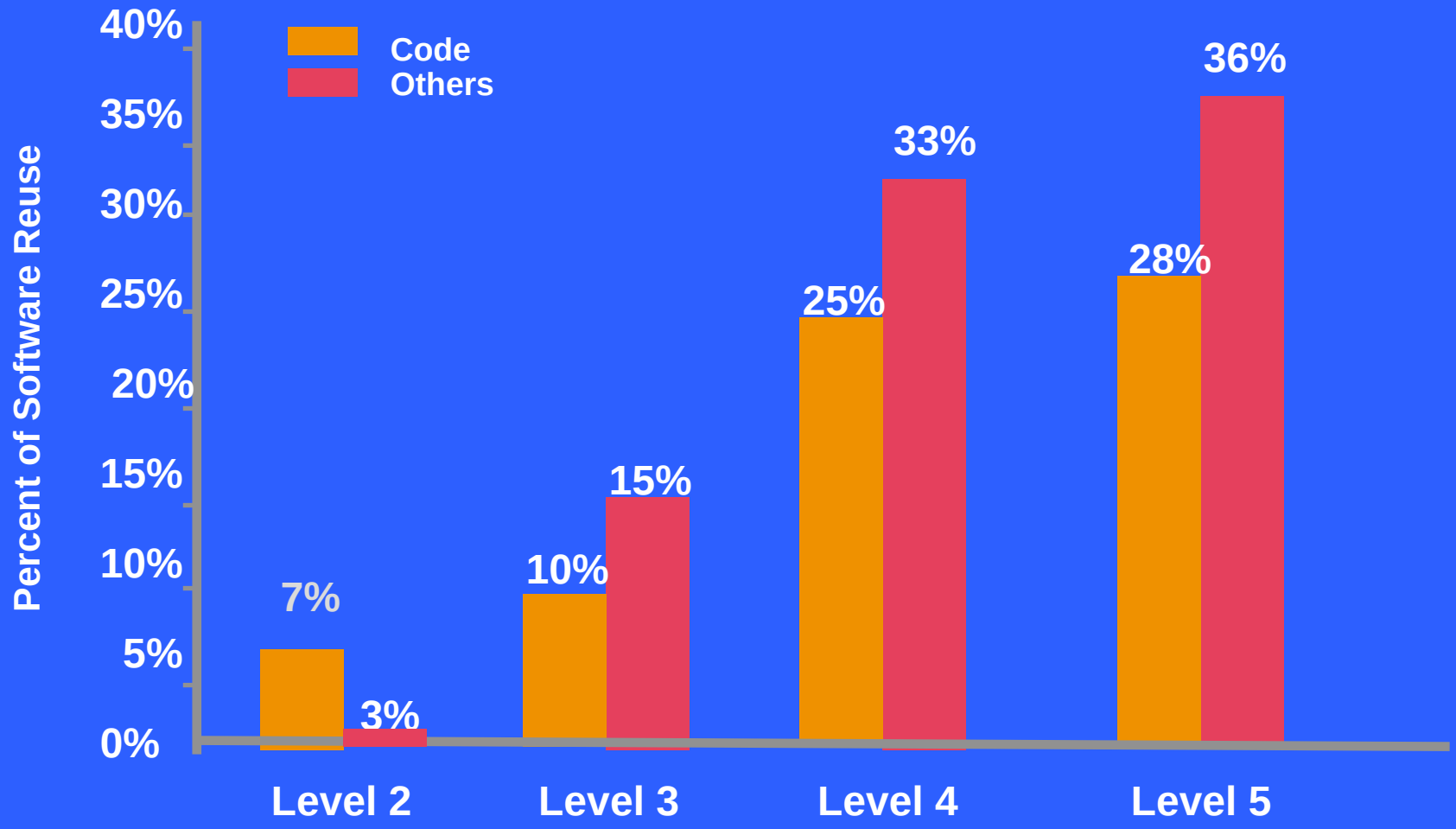


Reduce Cost = Increase Software Reuse



Based on 216 organizations assessment between 1991 to 2000

Software Reuse



Code reuse: No modification
Other reuse: Templates, Test Cases etc.

Some People Still Believe that...

Concurrent development of application software with target software, operating system software, or compiler software is well within our capability, and it will be no problem.

The impacts of an ill-conceived parallel development can be substantial.

Unexpected rework can have significant impacts on cost and schedule.

Failure to recognize the impacts of changes in one system could affect others, such as a small change to an operating system can create problems in all applications.

However, The Fact Is...

Project Management must ensure that the planning, schedule and budget are adequate for concurrent development.

Develop a risk mitigation plan for concurrent development.

Evaluate and isolate application software from the hardware and operating systems as much as possible.

Ensure interdependencies between application software and other components (Hardware, OS, COTS) are understood and accounted for in the planning and scheduling.

Some People Still Believe that...

There will not be any significant effort to incorporate Commercial-Off-The-shelf software (COTS) into our system.

Most Commercial Off-The-Shelf (COTS) are usually proprietary designs and may be poorly documented.

Integration cost and schedule may be impacted due to unanticipated subtleties in COTS external interfaces and design.

As system requirements change, unanticipated COTS modifications may be required, which could impact schedule, resources and cost.

However, The Fact Is...

Work closely with the user and suppliers early on to define COTS issues and ensure that the COTS products support the software development plans and activities.

Review integration issues early on with users and suppliers to define COTS and support issues.

Obtain agreement with COTS suppliers on technical and non-technical support requirements.

Create a plan to track COTS suppliers' updates and upgrades, and incorporate them into the project accordingly.

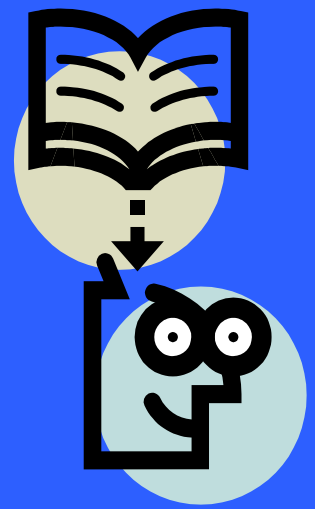
Some People Still Believe that...

Algorithm development is not a software problem; only technical people worry about algorithms.

Algorithms frequently turn out to be more complex than initially assumed.

Assuming that some existing algorithms can be reused without realistic evaluation.

Planning neither recognizes nor accommodates the need for algorithm development and verification.



However, The Fact Is...

Ensure that all algorithm requirements (including verification, usage conditions, response time, etc.) are defined and fully documented as part of the software requirements activities prior to formal design review.

Ensure algorithm developers are appropriately represented on the development team.

Establish completion criteria for algorithm definition and obtain commitment by the project team.

Define a risk mitigation plan if the physics behind the algorithms is not well understood.

Some People Still Believe that...

We do not have enough time and budget for integration testing; we can save time and money by putting all of the components together and test them all en-masse.

Belief that if all components are tested, integration should be easy.

Tendency to shorten the schedule and reduce the cost by minimizing integration testing.

Underestimate the complexity and efforts of integration testing.

However, The Fact Is...

Must plan and implement a well defined integration program.

Must ensure that the integration schedule has realistic allowances for testing, test failure, and retest time.

Review and monitor integration planning and adjust accordingly.

Ensure that all software components, which are part of the integration activities, have gone through all component testing and have satisfied the completion criteria.

Some People Still Believe that...

Software is so adaptable that small changes can be achieved without any significant impact.

Lacking knowledge of basic software project management.

Not recognizing the composite effect of numerous in-work changes.

Not following a disciplined process of cost and impact analysis.

However, The Fact Is...

Accept changes only after thoroughly analyzing the necessity and impact on the software project.

Ensure all changes are under software configuration control.

Establish software project management disciplines.

Ensure all changes are under strict control of software configuration management.

Understand the impacts of rework on completed components.

Promote the design of a software architecture with flexibility to accommodate changes.

Some People Still Believe that...

It is acceptable to select software subcontractors or suppliers solely on their reputation within the industry.

Lack of supplier selection criteria based on good technical and business sources.

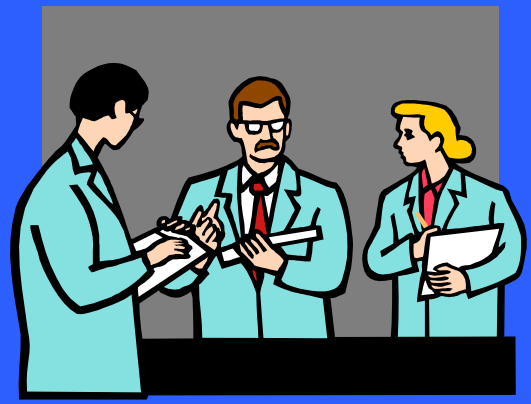
Assume today's capability of suppliers is the same as their past reputation.

However, The Fact Is...

Select subcontractors and suppliers based on their proposals, qualifications and past performance.

Define supplier selection criteria based on technical, business and past performance data.

Review supplier proposals to ensure their technical understanding of requirements.



Some People Still Believe that...

Software subcontractors and suppliers are dependable and do not need to be managed.

Many organizations do not monitor or manage software subcontractors.

Lack of communication between the project manager and subcontractor can lead to wrong assumptions.

However, The Fact Is...

Treat subcontractors and suppliers as an integral part of the team and manage the subcontracted product development as if it was another component of the organization.

Integrate subcontractor personnel with the project team, where appropriate.

Manage the subcontractor's performance from start to finish.

Conduct frequent reviews, and use metrics to monitor subcontractor performance.

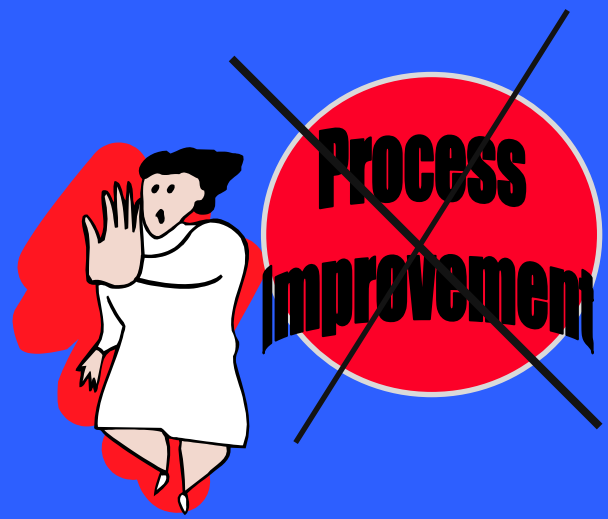
Some People Still Believe that...

Process improvement can wait until we have time, and when our people are not busy.

Many managers are too busy to address critical issues or take the improvement activities seriously.

Process Improvement is not applied in the context of the business but perceived as an additional burden.

A clever “resistance to change” tactic to postpone improvement indefinitely.



However, The Fact Is...

If you do not do it now – when will you improve ?

Software process improvement is critical to the success of the business.

Process improvement is the key discriminator of who will survive in these fast changing times.

If you do not improve, your competitor will…

10 Reasons for Process Improvement Failure

1. No long-term management commitment
2. Lack of experience and skill in process improvement
3. No clear vision of the desired results
4. Action plan too ambiguous (Ready! Fire! Aim!)
5. No quantitative feedback on progress (No measurements)
6. Wrong interpretation of the CMMI
7. Too many meetings & learning disabilities (Not walking the talk)
8. Wrong culture (Failed several times in the past, fear of change)
9. Wrong people in SEPG (document and planning to perfection)
10. Not everybody participating in the change process (ownership issue)

Five Keys to Look For…

1. Can the organization demonstrate actual business ***benefit*** of process improvement ? (improvement trends or r***esults?)***
2. Which projects ***follow*** (or not follow) the standard processes?
3. Are these processes being ***verified independently*** that they are used and controlled at the project level?
4. Is day-to-day ***decision making*** based on measurement data (where appropriate)?
5. How are business goals ***prioritized*** and intergroup ***conflicts*** resolved?

And …. The Last Key Point

So … your organization is appraised at CMMI Level 5 ….

Has Your Organization Experienced:

Business Improvement increases ?
Project Performance improvements?
Quality increases ?
Cost decreases ?
Customer Satisfaction Increases ?
Employee Satisfaction Increases ?

If NOT
Why?

My Conclusion

Is Your Oragnization Investing In Process Improvement For The Wrong Reasons?

Benchmarking Study Conclusion*

80% of businesses that fail to improve their software processes will fail by the end of 2006 (0.8 probability).

80% of businesses that do not achieve higher levels of capability maturity by 2007 will incur budget overruns of at least 50% (0.7 probability).

*** IEEE Software Industry Benchmarking Study 2001**

Conclusion

There are lessons to be learned to improve the way software and systems are developed and maintained.

There are several models (SW-CMM, CMMI, P-CMM, Six Sigma, ISO 9000 etc.) which organizations can use to improve their performance.

By commitment to the principles of continuous improvement, and actively managing those activities and utilizing "Lessons Learned" to avoid past mistakes, organizations can produce better quality products and achieve their business goals and objectives.

It’s Time To Get Out Of Process Improvement When…

- You decide to organize your household chores into a “Process Area”
- You teach your 3 year old child to spell “C M M I”
- You can eloquently explain the difference between “CMM" and “CMMI" to your Parent Teacher Association (PTA).
- You feel sorry for Dilbert's Boss.
- You insist on doing a “SCAMPI Appraisal" before you and your spouse produce another child.
- You ask the waiter what the restaurant's maturity level is.
- You use “Practices to Goal mapping” technique on your marriage proposal.
- You believe you never have any problems in your life, just “Process Improvement Opportunities."

BOEING®

Conclusion

Improving Software Development Practices Is Business 's Most Important Challenge

Questions & Answers