Bootstrapping Process Improvement Metrics: CMMI Level 4 Process Improvement Metrics in a Level 3 World

Jairus Hihn
Jet Propulsion Laboratory
California Institute of Technology

Scott Lewicki
Jet Propulsion Laboratory
California Institute of Technology

Scott Morgan
Jet Propulsion Laboratory
California Institute of Technology

Abstract[1]

The measurement techniques for organizations which have achieved the Software Engineering Institutes CMMI Maturity Levels 4 and 5 are well documented. On the other hand, how to effectively measure when an organization is Maturity Level 3 is less well understood, especially when there is no consistency in tool use and there is extensive tailoring of the organizational software processes. Most organizations fail in their attempts to generate, collect, and analyze standard process improvement metrics under these conditions. But at JPL, NASA's prime center for deep space robotic exploration, we have a long history of proving there is always a solution: It just may not be what you expected. In this paper we describe the wide variety of qualitative and quantitative techniques we have been implementing over the last few years, including the various approaches used to communicate the results to both software technical managers and senior managers.

1. Background

The Jet Propulsion Laboratory (JPL) in Pasadena, California is a US Government Federally-Funded Research and Development Center, which is run by the California Institute of Technology for the National Aeronautics and Space Administration (NASA). JPL's primary role is to build and operate unmanned, robotic space exploration missions throughout our solar system. JPL currently has 19 spacecraft and seven science instruments conducting active missions. All of these are part of NASA's objectives to explore Earth and space and to send robots and humans to explore the moon, Mars, and beyond. In any one year at JPL there may be more than 300 active software development tasks supporting these missions. Hence, even though JPL is known best for the unique hardware associated with robotic missions to Mars and other planets, the software written to support the operations of those missions is just as critical.

The successful completion of a JPL mission requires the development of software from every domain: the software that supports science instrument functionality, the flight software that controls the spacecraft, the ground software that sends commands to the spacecraft and instruments, the software that acquires and processes the engineering and science data sent back to Earth, the ground support equipment and the the software needed to test all of the software under development.

With each new mission, the amount of associated software and its underlying complexity has increased. This has caused the risks associated with the success of these missions to increase such that software is as important to mission as the hardware it runs on.

In response, the Software Quality Improvement (SQI) project was established at JPL in 2001 in response to the recognition of the need to improve mission software engineering practices across the Laboratory. An improvement strategy has been defined and executed based on industry best practices championed by the Software Engineering Institute of Carnegie Mellon University. Assessed at Maturity Level 3 in 2007 and having received internal investment funding for 8 years the JPL management and engineering communities increasingly want to know what is the impact of SQI? Has it been worth the well over twenty million dollar investment?

Different approaches to measuring impact have been considered and tried across the software industry [1, 2, 3, 4]. Return on investment measures (ROI) and quantitative impacts yield mixed results [5, 6]. ROI computations tend to require numerous assumptions, which are vigorously attacked in a mathematically

[1] The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

sophisticated engineering environment such as JPL. Furthermore, We find that traditional ROI computations and analysis do not work well in a not-for-profit setting. Traditional metrics such as productivity rates, defect density, cost growth can be tracked but they move slowly and are influenced by many factors, not just the process improvement programs [7, 8]. While waiting for the long term quantitative indicators to become available, short term indicators are needed.

CMMI Maturity Level 4 and 5 organizations are able to successfully use formal process control metrics to provide measures of the impact of their process improvement programs. Unfortunately, few Level 3 organizations are able to do this. Especially, organizations like JPL that permit projects to extensively tailor the organizational standard processes.

At JPL we are experimenting with various qualitative and survey based indicators to measure the software processes, the software products, and the software community that are feasible to implement in a Level 3 organization. The following sections provide an overview and summary of the specific approaches, along with examples of the results and their strengths and weakness. Specifically we will address achievable methods for Level 3 organizations like JPL for measuring process performance, for conducting customer surveys, tracking recommended process changes, obtaining and using standard project metrics and finally an approach for communicating the results. We have found that successful communication of what is known about the progress and success of the process improvement activities is just as important, if not more then obtaining accurate metrics. But first an overview of the current JPL standard development process will be described to provide context for the remaining sections on the implementation and use of metrics.

2. JPL Standards Overview

JPL has a set of defined policies and processes, which apply to all JPL employees and contractors, and all organizations, (project and line) developing or acquiring software, or integrating software subsystems, for NASA and reimbursable (non-NASA) efforts.

The policies are documented in the Software Development Requirements (SDR). The SDR is the JPL response to the NASA Procedural Requirements (NPR) 7150.2A. The Software Development Standard Processes or SDSPs translate the SDR into activities associated with specific process areas.

The SDSPs provide a lab-wide standard for implementing the SDR and are a collection of best practices for developing and acquiring mission software at JPL. They include work aids to help users perform their jobs more efficiently and effectively. The work aids consist of a large set of templates, handbooks, courses, seminars, compliance matrices, and various measurement repositories. The SDSPs were influenced by many sources: lessons learned; NASA requirements (NPR 7150.2, NASA STD 8719.13B); industry standards (CMMI-Dev, v1.2); and most important the JPL way of doing business. The SDSPs were first released in late 2006.

There are 22 individual SDSPs, see Figure 1 above. Each standard process is extensively documented containing

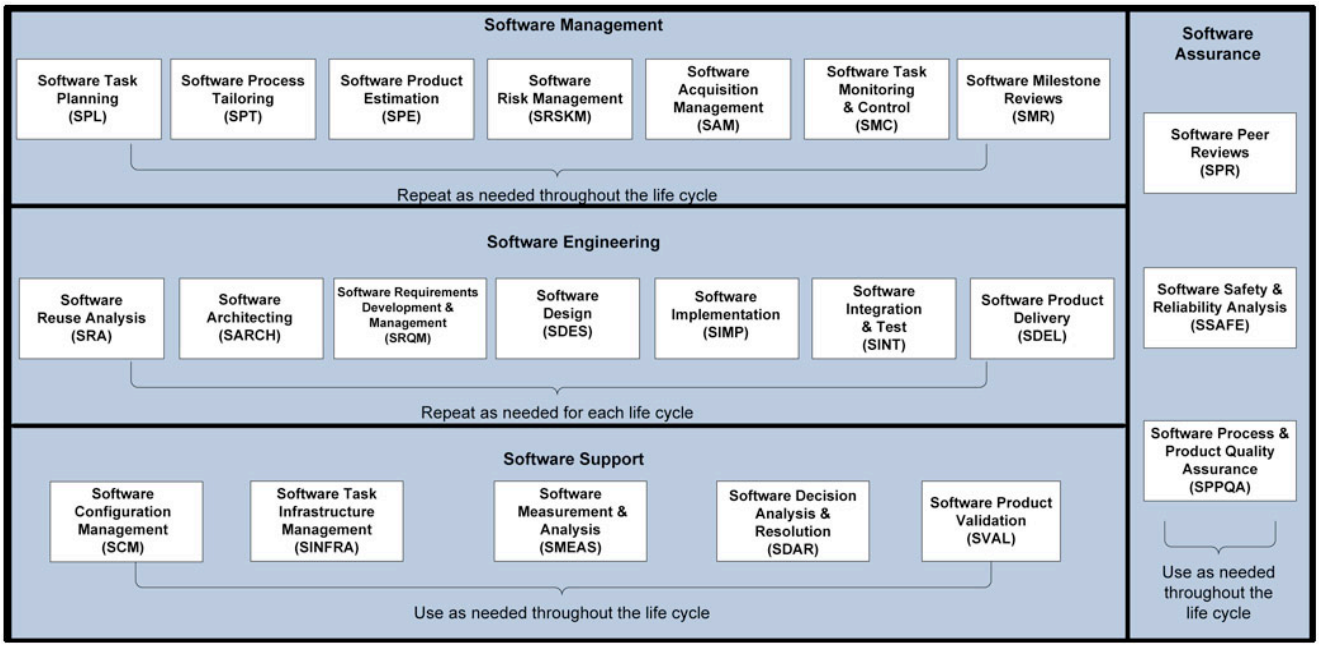


Figure 1: Overview of JPL Software Standard Processes (SDSPs)

- A 1-2 page overview describing the purpose, performing and supporting roles, usage scenarios, inputs and entry criteria, and outputs and exit criteria
- A context diagram that depicts the relationship of this process to the other SDSPs
- Detailed activity and sub-activity descriptions
- References and guidance material
- Appendices containing acronyms and abbreviations, a glossary, and a description of functional role responsibilities

Depending upon the project, not all processes may be applicable. For example, some projects may have no significant contractor developed software, and so the Software Acquisition Management process is not relevant. The process descriptions and activities are comprehensive standards and can be tailored to adjust to project-specific circumstances and requirements.

3. Process Performance

SDSP Tailoring Records

For the individual SDSP processes that are applicable, tailoring is performed following an established procedure and the resultant product is a Tailoring Record (TR) for that project or task. The TR compares the processes used on an individual software project to the institutional SDSP. The TR is generated early in the lifecycle of a project or task and ultimately assists the initial writing of software management and development plans.

The comparison captured in a TR is done sub-activity by sub-activity (relative to the SDSP) and the project or task supplies one the following responses:

A – Accept. Perform as stated in the SDSP.
M – Performed in a different Manner.
M – Performed informally.
M – Mostly performed.
M – Minimal implementation.
D – Not performed.
D – Not knowledgeable.
D – Not applicable.

The TR session itself represents a task "contact" and provides an opportunity for SQI to also provide recommendations to a project or task that are recorded in a Recommendations Tracking tool and can be followed up on.

Across multiple TRs for multiple projects and tasks, metrics can be obtained on which SDSP sub-activities generated more modified (Ms) and deleted (Ds) sub-activities. This can provide quantitative data on activities that projects and tasks may be struggling with. This can also provide information on areas where the SDSP itself can be improved.

Tailoring Record Review

The Tailoring Record then goes through an approval process called a Tailoring Record Review (TRR) that is also described by established procedures. TRR takes as input the TR and, for subsets of the SDSP sub-activities, maps the tailoring responses (i.e. the As, Ms, and Ds from above) to twenty-nine different Risk Areas commonly associated with projects and tasks. The Risk Areas include:

1) Are the task processes and practices sufficient to identify and record requirements?
2) Are the task processes and practices sufficient to evaluate the requirements for usability and completeness?
3) Do the task processes and practices effectively control requirements changes and evaluate the impact of the changes on the task's schedule and costs?
4) Do the tasks processes and practices include capturing the software design at a sufficient level of detail?
5) Are the task processes and practices sufficient to evaluate the software design for usability and completeness?
6) Will the task processes and practices effectively evaluate the quality, functionality, and performance of the software units?
7) Are the task processes and practices adequate to test the integrated software product?

…and other areas. For each of these twenty-nine areas an assessment is made whether the task's tailoring choices (as captured in the TR) represent an implementation risk. If so, a standard risk analysis is performed and the impact, likelihood and any possible mitigation is determined and recorded. In addition any unique implementation strengths are noted such that they may be conveyed to other tasks.

Just like with the TR sessions, the TRR provides another opportunity for SQI to provide recommendations to a project or task. These recommendations are recorded in a Recommendations Tracking tool and can be followed up on.

Software Process Review

As mentioned above Tailoring Records and their corresponding Tailoring Record Reviews are performed early in a project or task lifecycle. For on-going tasks (mid-lifecycle) or tasks that are transitioning into maintenance or closeout, a Software Process Review (SPR) is performed. The purpose of a SPR is to follow up with on-going tasks that have filled out Tailoring Records and have performed Tailoring Record Reviews (TRRs) in order to take a measure

("heart beat") of process performance on existing tasks.

The process was designed to fit what the CMMI model asks for in the Organizational Process Focus Process Area and is light-weight with minimal effort required of tasks.

It also provides another opportunity for "two-way street" dialogue with task:

- Tasks may be provided with information about new tools or methods to help their task's performance
- Process owner and SQI get additional feedback on processes that can be incorporated into SDSPs, work aids, and other assets.

The SPR asks the question: based on the risks, strengths, recommendations identified at the planning stage, "How are things working for you?"

Specifically:

- Are your processes effective? (That is, did you accomplish the process objectives?)
- Have any processes been descoped?
- Are your resources adequate? (That is, were resources adjusted significantly different than plans?)

The responses to these questions (including any additional risks, recommendations, or strengths) are recorded in a form. Each SPR may also produce recommendations, which are captured in the Recommendations Tracking tool.

Work Product Checklist

A Work Product Checklist (WPC), associated with each of the software products and projects, captures the types of documentation that were generated and the tools that were utilized on the project. The WPC is a set of approximately 60 questions derived from the Tailoring Record that provides a quick look at process performance based primarily on the existence of key products. It is updated at least every three years.

The WPC is a primary input into the JPL State of Software Process Chapter (Discussed in Section 6). The WPC is used to identify low performing activities as candidates for process improvement and to support SQI strategic planning. The WPC is also used as input into the SPR.

The sample of software tasks that are requested to complete a WPC is determined by the following steps:

1. Identify all Software Inventory tasks in the DSP database that either have no WPC or the WPC is greater than 2 years old.
2. If the identified products that require a new or updated WPC do not stratify the population, then random select products to meet stratification criteria
3. Stratification criteria are that there must be at least 5 products for all combinations of SW class vs. Team Size.
4. Archive the results in the database described below.

Stratification is performed on years when a comprehensive WPC collection is not performed so that the set of tasks requested to complete WPCs are representative of the actual laboratory population of tasks.

DSP Database

All of the data collected by the methods described above are stored in the Develop Software Products (DSP) Database. In addition the DSP DB contains the software inventory. The Software Inventory, conducted every two years, is a comprehensive survey of all mission related software on the lab. In the future, the DSP DB will contain the responses from the SPR's and other deployment and customer outreach activities. The data can be input by either web-based forms or en-mass from formatted spreadsheets.

The DSP DB enables the mapping of records and data items across the various data sources, and provides various views and reports for examining the data. In addition, each view can be exported into spreadsheets for analysis.

Process Performance Metrics

One of the most effective things from a metrics standpoint we have been able to derive are measures of process performance based on the detailed information collected from the TRs and WPCs. The term process performance is used to indicate the percentage of SDSP activities performed as defined or in an approved modified form for a task or group of tasks Figure 2 displays the aggregate results from 2008. As expected Class B[2] tasks, which have a higher required reliability have a higher process performance as indicated by the 84% for activities performed. In the current JPL environment where older tasks have been grandfathered and a history of process 'flexibility' greater then 80% is considered high performing. As the new processes become more widely established the standard for high performing will be increased. A major objective of measuring process performance is to be able to track this transition. It is also not surprising that tasks that completed a TR have a higher level of performance as these tasks were hand picked based on the perception that they were higher risk tasks due to mission criticality or budget exposure. Therefore, they

[2] Class B software is mission critical and Class C software is mission support. Failure of Class B software can cause significant loss in science return.

were more closely monitored and expected to follow the process standards.

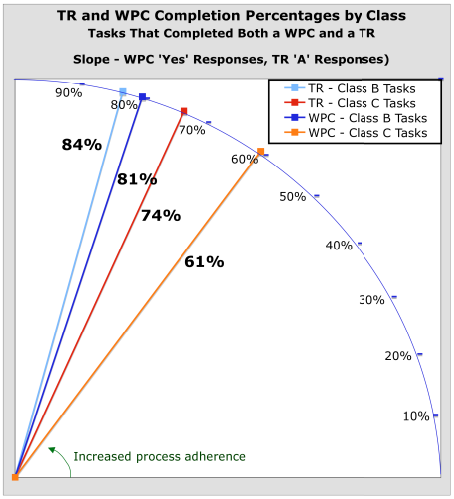


Figure 2. WPC and TR Completion Rates by Software Class

Another important outcome of the extensive process monitoring is the ability to clearly identify low performing activities especially as related to the risk areas used in the TRRs. The results from 2008 are displayed in Table 1. PPI is Process Performance Index and is percent of sub-activities performed in an approved manner. As can be seen these spread across both technical and management processes.

Eleven Least-Performed Risk Area Activities		
Process	Activity	PPI
Design	Develop the detailed design.	75%
Implementation	Review the developing code.	85%
Integration	Perform regression tests as needed.	85%
Management	Provide inputs to institutional Software Process Management (SPM) process.	67%
Management	Organize the software management function.	78%
Management	Manage risks.	83%
Planning	Estimate the effort and cost of the software project/task.	87%
Quality Assurance	Objectively evaluate the project's activities against the planned processes.	84%
Requirements	Manage software requirements changes.	85%
Requirements	Formally review and approve the software requirements.	87%
Trade Studies	Develop and Assess Criteria.	82%

Table 1. Least Performed Risk Area Activities

Deriving Process Improvement Impact Metrics

In 2009 each major element of SQI was required to define at least one "Impact Metric" that would be tracked and reported throughout the coming year. Each Impact Metric is meant to provide a quantitative measure of SQI's positive impact to software tasks at JPL. As the TRRs and the SPRs were a major area where SQI broadly interacted with the software community, these were selected to be the areas that would be measured. As process change ripples through the organization slowly, this also meant that the recommendations tracking had to play a key role for deriving short term impact indicators.

At JPL impact metrics are required to be defined using a specified method based on documented success criteria and numeric goals.

- **Success Criteria** - TRRs and SPRs are shown to be effective at evaluating task practices to identify strengths, weaknesses, and risks.
- **Numeric Goals** for TRR and SPR performance:
  - 100% of the TRR recommendations and risks should be recorded in Recommendations Tracking Tool
  - 66% of the identified recommendations should be at least somewhat useful

Examples are displayed in Figures 3 and 4 above where it can be seen we are currently below our goals but we also know what we are working towards.

Tailoring Record Review Impact Metrics			
	Goal	Feb	Apr
TRR Recommendations and Risks		87	90
Total Recommendations Recorded		76	80
Percent Recommendations Reported	100%	87%	89%
Open		20	19
Percent U/SU	66%	47%	46%
Usefull/SU		22	23
Not Useful		25	27

Figure 3. Example Tailoring Record Review Impact Metrics

Software Process Review Impact Metrics			
	Goal	Feb	Apr
SPR Recommendations and Risks			9
Total Recommendations Recorded			8
Percent Recommendations Reported	100%		89%
Open			8
Percent U/SU	66%		0%
Usefull/SU			0
Not Useful			0

Figure 4. Example Software Process Review Impact Metrics

4. Customer Feedback

SQI interactions with the software community are tracked at various levels including customer feedback surveys, focus groups and recommendations tracking.

In the commercial sector especially the service industry collecting customer feedback is the norm. Customer surveys are less common but still frequently used in engineering companies. Our survey is done annually on-line using ZIP Survey[3]. The strength is that they are easy to administer with a 50% return rate. We find that customer surveys provide fuzzy feedback with respect to the community's actual perception as to usefulness of SQI's products and services. The one form of feedback that comes through clearly is when members of the software community have strong

[3] Zip Survey is a tool for design web based survey forms. See http://zipsurvey.com.

negative feedback. Even then one must be careful not to over react to a squeaky wheel.

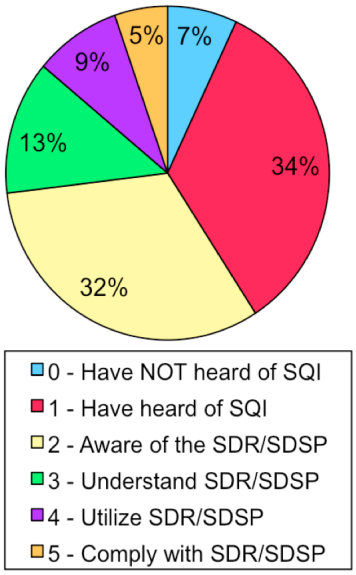


Figure 5: Software Community Awareness

On the other hand, customer surveys provide reliable information on basic awareness of the key aspects of the process improvement program. Especially, when questions are specific and when additional questions are added to catch inconsistent responses. Figure 5 above is the summary results from the 2009 survey and Table 2 displays examples of the specific questions. These provide a key baseline from which improvements in awareness can be tracked. The responses shown in Table 2 are for line managers who are responsible for over seeing the implementation of JPL processes and the results indicate only 50% were aware of the policy level requirements and a little over a third knew the standard processes existed. Both of these indicate there is significant room for improvement.

While these metrics are useful for measuring awareness and the process improvement program has penetrated the software community it does not indicate whether the process improvement activities are really an improvement. As a next step in working with qualitative indicators of impact a procedure and tools were implemented to enable obtaining and documenting customer reported impacts resulting from interaction with SQI. For better or worse we call this 'recommendations tracking', and SQI has developed an in-house Recommendations Tracking tool to support this effort.

Question	Yes	No
Are you aware of the requirements in the Software Development (SDR) requirements document (JPL DocID 57653) and how they apply to the software management, software development, and software acquisition activities in your organization?	50%	50%
Are you aware of the tailorable processes in the Software Development Standard Processes (SDSP) standards document (JPL DocID 74352) which aid in implementing the requirements in the Software Development Requirements (SDR)?	36%	64%

Table 2: Sample Awareness Questions

5. Recommendations Tracking

A recommendation, in this context, is an actionable suggestion to software task regarding their products or processes. However, any service that requires more than 2 hours - even if no recommendation was made still is tracked in order to determine if the service provided had value.

SQI has a saying "if it is not written down then it did not happen." The documented recommendations procedure is displayed in Figure 6 below. Here it can be seen that all contacts are recorded, and then there is further follow up, if there is the possibility of an impact (positive or negative). Figure 7 contains examples of the forms used for recommendations tracking.

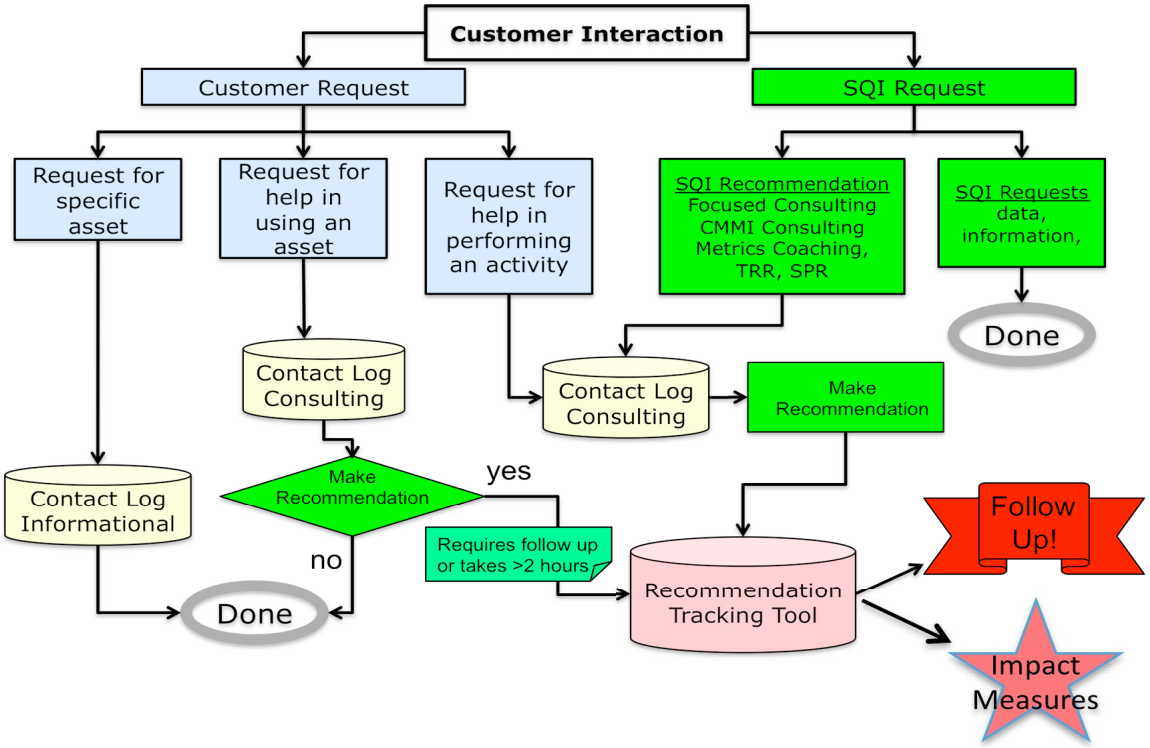
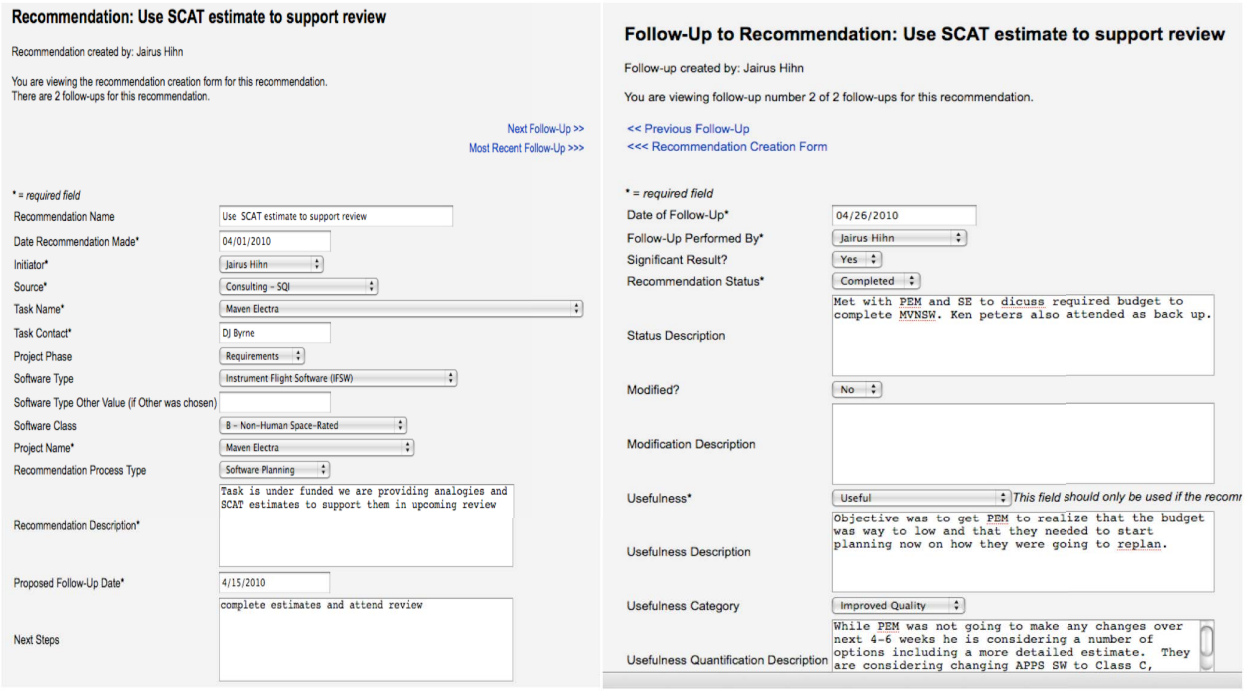


Figure 6: Recommendations Tracking Procedure



Recommendation: Use SCAT estimate to support review

Recommendation created by: Jairus Hihn

You are viewing the recommendation creation form for this recommendation.
There are 2 follow-ups for this recommendation.

Next Follow-Up >>
Most Recent Follow-Up >>>

* = *required field*

Recommendation Name	Use SCAT estimate to support review
Date Recommendation Made*	04/01/2010
Initiator*	Jairus Hihn
Source*	Consulting – SQI
Task Name*	Maven Electra
Task Contact*	DJ Byrne
Project Phase	Requirements
Software Type	Instrument Flight Software (IFSW)
Software Type Other Value (if Other was chosen)	
Software Class	B – Non-Human Space-Rated
Project Name*	Maven Electra
Recommendation Process Type	Software Planning
Recommendation Description*	Task is under funded we are providing analogies and SCAT estimates to support them in upcoming review
Proposed Follow-Up Date*	4/15/2010
Next Steps	complete estimates and attend review

Follow-Up to Recommendation: Use SCAT estimate to support review

Follow-up created by: Jairus Hihn

You are viewing follow-up number 2 of 2 follow-ups for this recommendation.

<< Previous Follow-Up
<<< Recommendation Creation Form

* = *required field*

Date of Follow-Up*	04/26/2010
Follow-Up Performed By*	Jairus Hihn
Significant Result?	Yes
Recommendation Status*	Completed
Status Description	Met with PEM and SE to dicuss required budget to complete MVNSW. Ken peters also attended as back up.
Modified?	No
Modification Description	
Usefulness*	Useful *This field should only be used if the recomm*
Usefulness Description	Objective was to get PEM to realize that the budget was way to low and that they needed to start planning now on how they were going to replan.
Usefulness Category	Improved Quality
Usefulness Quantification Description	While PEM was not going to make any changes over next 4-6 weeks he is considering a number of options including a more detailed estimate. They are considering changing APPS SW to Class C,

Figure 7: Sample Recommendations Forms

As Figure 8 shows, a Customer Interaction with SQI can have several flavors and can follow several paths. One type of Customer Interaction involves the promotion (push) by SQI of a particular tool or technique. This is called Focused Consulting. Another type of Customer Interaction involves tasks or individuals requesting help (pull) from SQI. This is called Ad-hoc or On-Demand Consulting. Each of these follows a specific path in the Recommendations Tracking Procedure in how the "contact" is recorded and the recommendation is captured and processed.

The following are a number of scenarios/examples to clarify how this procedure is made operational:

1. Only the initial "contact" is counted, hence it is one contact even though there may be multiple meetings.
2. Focused consulting
   - A recommendation is made to use a tool, such as Coverity (a static code analyzer), in a one minute conversation and they tell us to take a hike or they are already using it, it is a consulting contact.
   - If they agree to meet and hear about the tool, it becomes a recommendation.
     - If they decide they are not interested, it is a Recommendation that will be scored 'Not Useful' and the SQI representative will try to find out why.
     - If they decide to try the tool (e.g Coverity), then the SQI representative records this as a recommendation and sets a date for following up. Most likely this will end up as a 'Useful' recommendation, but maybe not.
3. On-Demand Consulting
   - Someone requests a cost estimate which is completed in 1 hour
   - In this case, record as a contact and as a recommendation. This is so that the SQI representative makes sure to follow up whether the estimate was used and what made it useful or not useful.
   - Yes, it took less then 2 hours. But that is a guideline and the real issue is whether the service provided is worth following up on.

The customer responses are captured in the Recommendations Follow Up Forms (see Figure 7). This enables the ability to generate metrics reports as shown in Figures 8 and 9, which are used in monthly and quarterly status reports. Figure 8 indicates that 24% of the recommendations are found to be not useful by the software community and 30% are found to be only somewhat useful. "Not Useful" can indicate a variety of issues which require follow up. "Not Useful" can mean that the activity or work aid being recommended needs to be modified or deleted (i.e. institutional processes need to change) or it can indicate a lack of understanding or out and out resistance to change. In the later case the task team needs to change their behavior but the initiator still has to be SQI and we need to determine how to better engage the task.

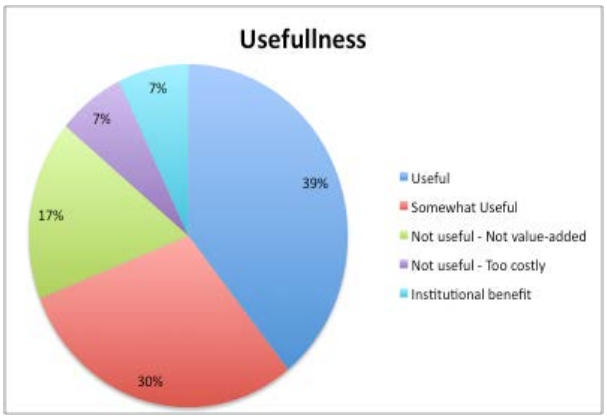


Figure 8: Percentage Completed Recommendations by Useful and Not Useful Categories

Recommendations arise from various sources. TRs, TRRs, and SPRs, On-Demand Consulting and Focused Consulting. All of these have been described in varying detail in the text above. The one other major source of recommendations is during CMMI Assessment preparation activities. In this case, as weaknesses are identified based on the CMMI process model, required and recommended changes are identified.

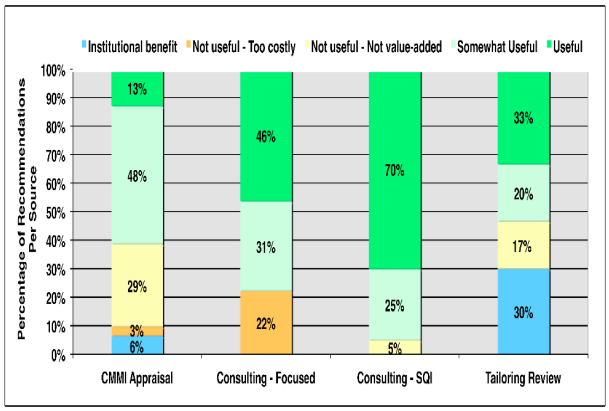


Figure 9: Percent Useful/Not-Useful by Recommendations Source

The data in Figure 9 is used to understand what types of community interactions generate more or fewer useful recommendations. Not surprisingly when someone comes to SQI for help and their specific needs are being addressed, the percentage of useful

responses are well above average. Next, and also not surprising, engineers are more open to recommendations for tool usage and least positive concerning "process" related recommendations. This information is being used in a variety of ways, but most importantly, it is used to create a baseline from which to measure changes and to focus SQI on identifying ways to improve the effectiveness of process related recommendations.

The strength of tracking subjective feedback for the JPL software community is that a large quantity of data can be obtained in a relatively short time period. We currently have generated 330 recommendations in 8 months with 265 closed. These are currently being used to derive specific impact metrics which will be described at the end of the next section.

The weakness is that the data is subjective and subject to the human biases of the SQI staff and the software engineers. For example, when an engineer says something is "Somewhat Useful," they can mean many things from what the words literally mean to "I am talking to you on the phone, and unless I really hate it, I am not going to be confrontational". This means this type of response is very noisy. Another issue is that the SQI staff is not following through on trying to obtain estimates of the quantitative impact or pointers to actually quantify the data. This latter issue is in the process of being addressed through improvements in the tracking tool and additional training and monitoring of the reports.

6. Project Metrics

Projects and tasks are tracked using two approaches, by contributing information to a SQI-maintained Software Inventory and by collecting a standard set of Milestone Metrics.

The Software Inventory is an established and maintained list of software products and projects at JPL which includes data on software classification/criticality, implementation status, lines of code count, primary and secondary languages used, effort in work years, and other characteristics. It is conducted broadly across the entire engineering part of the JPL. In 2007 we identified 38 million lines of code being developed or maintained by 228 tasks.

Figure 10 displays how tasks are evolving over time. However, some of the difference is due to improved reporting as the organization becomes use to systematically providing such data.

Milestone Metrics are a detailed set of planned and actual metrics collected for JPLs major missions. Over the years JPL has increasingly realized the need to be able to make quantitative-based decisions at both the strategic and tactical management levels. The response was to implement a software metrics system. After 8 years of data collection we are starting to be able to extract some quantitative indicators of the impact of becoming CMMI Maturity Level 3. In Figures 11 and 12, it can be seen that projects that follow a disciplined process have higher productivity rates and lower effort growth then other projects. We currently do not find the differences to be as dramatic as reported in the literature, but they are significant and these results do receive significant management attention

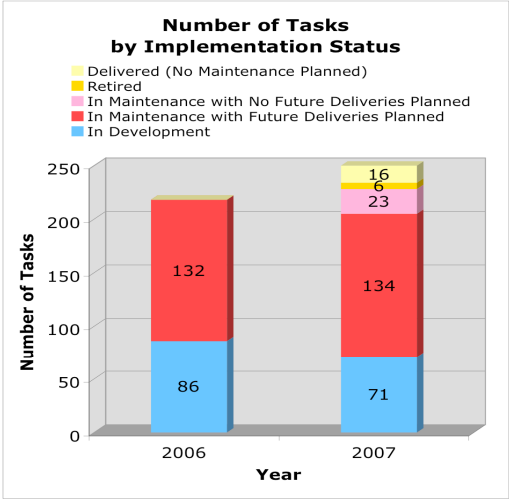


Figure 10. Number of Tasks by Implementation Status by Year

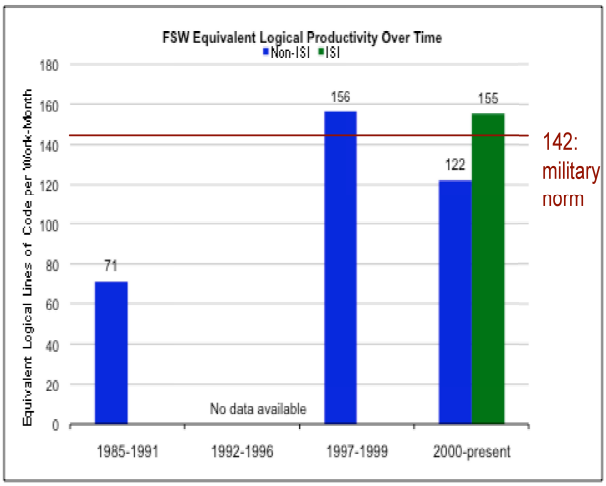


Figure 11. Spacecraft Flight Software Productivity Rates by Year

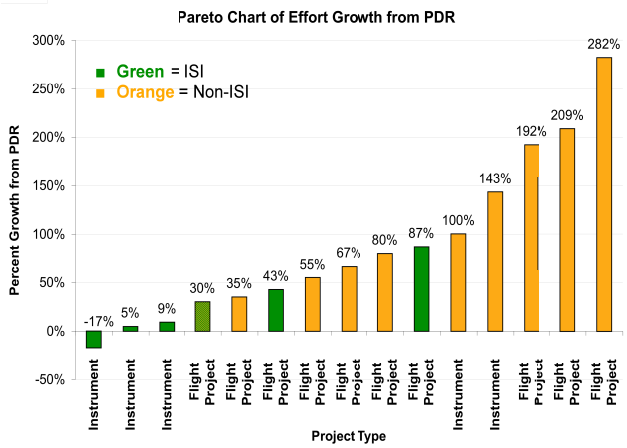


Figure 12. Effort Growth from PDR

7. Communicating the Results

Great metrics not communicated will have no impact on an organization and 'soft' numbers well communicated can have a significant impact. Because of this the JPL SQI project uses a number of different mechanisms to communicate what our data is telling us about JPL and to promote increased use of metrics across our projects and tasks. Once a month, Software Engineering Process Group meetings are held within each division, and noontime briefings that are broadly advertised are held. In these meetings the products, services, standards, and metrics developed and maintained by SQI and by individual contributor are discussed. The SQI project manager meets regularly with the Section managers of the major software sections to make certain these key players in the software community are made aware of where JPL stands and where their section stands relative to organizational averages.

In 2008 JPL made an innovative step forward and developed its first State of Software Report [9]. This 120 page report summarizes and analyzes virtually every piece of data that the we have collected over the previous years. It describes in detail the state of JPL's software products, processes and people. Some of the data is very mature and some is quite noisy but taken together it provides a comprehensive picture. A number of the more interesting charts shared in this paper come form this report and it is quoted widely in various training classes and briefings. We have found getting things in writing in a widely distributed document has an impact in and of itself.

8. Conclusion

Most companies, if they cannot do it 'right', give up on metrics programs. At JPL we do not know how to give up as demonstrated by the many successful, one of a kind missions that have been completed over the years. This same 'can do' attitude has enabled us to boot strap a wide variety of approaches in the face of numerous barriers to standardization.

As a CMMI Maturity Level 3, and highly decentralized, organization that is still learning to make it real, JPL has experimented in and succeeded in establishing an organization wide metrics program that provides both qualitative and quantitative ways of accessing the impact of the software improvement activities over the last few years. These metrics include multiple dimensions and are from sources that are both subjective and objective. But they ultimately provide a useful means of steering the direction of future process activities and demonstrate the value of process improvement.

9. References

[1] Anandasivam Gopal , M. S. Krishnan , Tridas Mukhopadhyay , Dennis R. Goldenson, Measurement Programs in Software Development: Determinants of Success, IEEE Transactions on Software Engineering, v.28 n.9, p.863-875, September 2002.

[2] Tracy Hall , Norman Fenton, Implementing Effective Software Metrics Programs, IEEE Software, v.14 n.2, p.55-65, March 1997.

[3] Shari Lawrence Pfleeger, Lessons Learned in Building a Corporate Metrics Program, IEEE Software, v.10 n.3, p.67-74, May 1993

[4] C. A. Dekkers, "The secrets of highly successful measurement programs," Cutter IT Journal, vol. 12, pp. 29--35, 1999.

[5] D. Galin, M. A. Lipman, Are CMM Program Investments Beneficial? Analyzing Past Studies, IEEE SOFTWARE , November/December 2006, pp 81-87

[6] Rini van Solingen, Measuring the ROI of Software Process Improvement, IEEE SOFTWARE, May/June 2004, pp 32-38.

[7] Daniel J. Paulish , Anita D. Carleton, Case studies of software-process-improvement measurement, Computer, v.27 n.9, p.50-57, September 1994

[8] J. Iversen, L. Mathiassen, Lessons from Implementing a Software Metrics Program, Proceedings of the 33rd Hawaii International Conference on System Sciences, Maui, Hawaii, January 04-07, 2000, p.7040

[9] R. Gostelow, J. Hihn. et. al. JPL Mission Software: State of Software Report 2008 External Release, JPL D-29114, 2008. Contact jhihn@jpl.nasa.gov for a copy.