Use of Metrics in High Maturity Organizations

Pankaj Jalote
Department of Computer Science and Engineering
Indian Institute of Technology Kanpur
Kanpur, India – 208016
Jalote@iitk.ac.in

Summary

A high maturity organization is expected to use metrics heavily for process and project management. A study was conducted to understand how some of the high maturity organizations use metrics. This report summarizes the similarities found in the use of metrics, focusing on the metrics infrastructure employed, use of metrics in project planning, use of metrics in monitoring and controlling a project, and use of metrics for the improvement of the overall process.

Introduction

Software process improvement (SPI) has emerged as a critical area for an organization involved in software development. Various organizations have reported benefits from software process improvement programs [Art97, But95, Hum91, Dia97, Dio93, Hal96, Hol97, Lip93, Woh93, Woh94], and now there is little doubt that process improvement can pay rich dividends.

For SPI, currently perhaps the most comprehensive framework and most influential frameworks if the capability maturity model (CMM) for software [Pau95]. The CMM classifies the maturity of the organization in five levels – level 1 through level 5, with level 5 being the highest. For the purposes of this report, we will consider level 4 and 5 as the high-maturity levels. The number of high maturity organizations has been increasing rapidly over the years and from less than 20 a few years back, the number has grown to over 70. In high maturity organizations metrics are expected to play a key role in overall process management as well as in managing the process of a project.

As the CMM framework is not prescriptive, it does not specify which metrics should be used or how they should be used. So, there is a possibility that different organizations may employ different metrics and use them differently. However, as the basic objectives of high quality, high productivity, and small cycle time are same in all organizations, and because of the common history of metrics and standard practices in other disciplines, there is a good chance that there will be similarities in the approaches and metrics used. A general study regarding practices of high maturity organizations found that many similarities do in fact exist in high maturity organizations [Pau99].

The aim of this study is to see how similar are the metrics programs of high maturity organizations, and the nature of the similarities. One of the main problems in software process improvement initiatives is not knowing clearly what to do, as was revealed by an SEI survey [Her96, Her97]. At high maturity organizations, metrics is expected to play a key role. If nature of metrics programs and similarities between them are determined, then it can help other organizations in building or improving their own metrics program and in their quest for high maturity. Providing this input to the community is the primary objective of this study.

Software metrics and measurements have been an area of active interest for a long time. One of the main objectives of the area has been to quantify properties of interest in the process or the products, with the goal that these can be used to evaluate and control the products and processes. Though metrics can be used in many ways, in a software organization, the three main uses of metrics data are: for project planning, for monitoring and controlling a project, and for overall process management and improvement. This study focused on these three uses of metrics. To support these uses, some metrics infrastructure is needed. This study also considered the metrics infrastructure in high maturity organizations. These four areas form the core of this study.

For conducting the study, a few high maturity organizations were selected. The author visited these organizations personally to understand the role and use of metrics. The information collected during these visits was the main source for preparing this report. The study found that most of the organizations studied collect similar metrics and have similar metrics infrastructure in place, though the details of the procedures followed in the use of metrics differ somewhat. In the rest of the report we discuss further details about the similarities in the use of metrics in these organizations.

Methodology

A questionnaire was first prepared, which listed the key questions regarding metrics and their usage. The questions regarding metrics were grouped in four categories – about metrics infrastructure, use of metrics in planning, use of metrics in managing a project's process, and use of metrics in managing and improving the overall process. Besides these, there were some general questions about the organization, its SEPG, etc., and some other miscellaneous questions.

The author visited the organizations personally, discussed the questions with some senior members of the SEPG of the organization, and sought clarifications, where needed. The questionnaire was then filled by the author. Each organization was given an assurance of complete confidentiality. The filled questionnaire was then sent to respective organizations for validation, and were later used for this report.

As the possibility of subjectivity is more at higher maturity levels, interpretation of the lead assessor becomes important. And different lead assessors "interpret" some of the requirements at higher levels in a different manner (for example, at level 4 some assessors "require" statistical process control, while others do not view it as essential). To ensure that the outcome of the study is not biased by the interpretation of some assessors, the organizations chosen for the study were those which had different lead assessors. By doing this, we "randomize" the "assessor factor".

All the organizations that were picked are in India, though some of the organizations are development centers of multinationals based in the US. There are now many high maturity organizations in India (by some estimates about half of the level 4 or 5 organizations in the SEI database are from India). The primary reason for selecting the organizations from India were author's contacts and access, and the ease and low cost of conducting the study. However, selecting organizations from one country may bring in a "cultural" bias.

All the organizations studied were in the software services sector. That is, they executed projects for some customers. Though some of the projects in these organizations were to develop or maintain general-purpose products for the customer, for the organization studied, the software development activity was primarily a project driven by customer requirements. The number of software engineers in the organization varied from about 250 to about 3,000 (if only a part of the organization was assessed we are considering only the strength of that part). All except one organization were ISO certified, with ISO certification preceding their CMM assessment. That is, all except one seem to have adopted the ISO framework first, and have later moved to the CMM framework. All the ISO certified organizations continue to maintain their ISO certification.

Metrics Infrastructure

All organizations had a formally defined unit that was dedicated to, and was responsible for, SPI. This unit is frequently referred to as the software engineering process group (SEPG). Most organizations had the SEPG strength of less than 2%. Only one organization had the strength of SEPG as 6%, but for this organization the SEPG also did tool and technology development, which no other SEPG did. The main activities of the SEPG generally were, quality assurance (verifying that the processes are being followed), training, process analysis and definition, and consultancy. In most of the organizations, for the assurance work, the SEPG played the role of a coordinator, while the actual work of audits was done by people from other parts of the organization. Similar was the situation with process definition – SEPG plays the coordinator role, while some task force does the actual definition. Generally, the maximum effort of the SEPG was spent in process deployment related tasks (audits, training, providing help, etc.)

Having an SEPG provides the necessary people support for processes-based improvement in the organization. Clearly, having an SEPG is not sufficient. Since a foundation of the process-based approach for software development is that the past performance of a process can be used to predict performance on a project, it follows that data on past performance must be maintained for planning a project. Process database and the process capability baseline form the two key infrastructure elements for making past performance available for use [Jal99].

We consider a process database in an organization as a database (or a collection of databases) that contains historical information and data about the use of organization processes on (completed) projects. The CMM requires that the organization have a "process database", which is used for planning, though the CMM does not specify what this process database contains. All the organizations maintain a process database – some have an integrated database while others kept performance data in multiple databases (e.g. the review data might be maintained in a separate database). The process database, invariably captured information about:

- Project size
- Effort
- Schedule
- Defects
- Risks

For size, the data generally kept was estimated size and actual size. Different units for size were used, with LOC being the most common, though function points were also used. "Components" as a unit of size was also used. Some organizations worked with multiple units. For effort, generally total effort, and effort spent in different phases was kept. Rework effort was also captured (to help determine the cost of quality). Unit for effort in the database varied from person-hours to person-months, though most organizations captured effort in hours in a project.

For defects, the total number of defects and distribution of defects with respect to where they were detected was captured. Distribution by severity, category, cause, etc. is also frequently recorded. Most also maintain data about the origin of defects, which helps in computing the defect removal efficiency of various quality control tasks. With origin and detection stages, the defect data for a project becomes a two-dimensional table (an example can be found in [Jal99]). For schedule, the main data maintained was the actual and estimated dates so slippage can be computed. For reviews, effort and defect data was generally recorded separately for self-preparation and the review meeting. This made detailed analysis possible.

For risks, generally the risks identified by the project during planning, and the risk perception at the end of the project were both captured. Most also maintained information about risk mitigation strategies. Besides these, some organizations kept different data like number of change requests, number of baselines, number of risks, number of reviews, etc.

All organizations determined their process capability from the past data (it is a goal at level 4 of the CMM). The capability of a process specifies what is the range of expected outcomes if the process is followed. A capability baseline is essentially a snapshot of the process capability at a time. By computing capability baselines at regular intervals, the trends in process capabilities can be analyzed. Understanding the capability of the process and seeing the trends is one of the main purposes of the capability baseline in most organizations. Some organizations also used it to set overall process improvement goals, while some also used it for planning purposes. Most computed the capability of overall process in terms of productivity, quality, distributions, removal efficiencies, etc., while some also computed percent overrun in cost and schedule, percent of defects found before shipment and the percentage found after shipment. Capability of component processes was also frequently computed, particularly for the review process.

Organizations specified their capability in various ways. Some used the classical definition of 6-sigma around the mean, while others specified it as the spread of the past performance data, or simply as the mean and standard deviation. Also, for most characteristics like quality, productivity, the spread of the data was quite large. Some defined "focussed" baselines to have smaller spread in the values.

Clearly the process capability baseline can be computed from the data in the process database. And where does the data in the process database comes from? As the data in a process database is about completed projects, this data comes from a postmortem analysis or a closure analysis of the project. This is the analysis that is done on the project after the delivery has been done. It is an excellent tool for learning and building a knowledge base. The relationship between the process database, the process capability baseline, and closure analysis is shown in Figure 1 [Jal99].

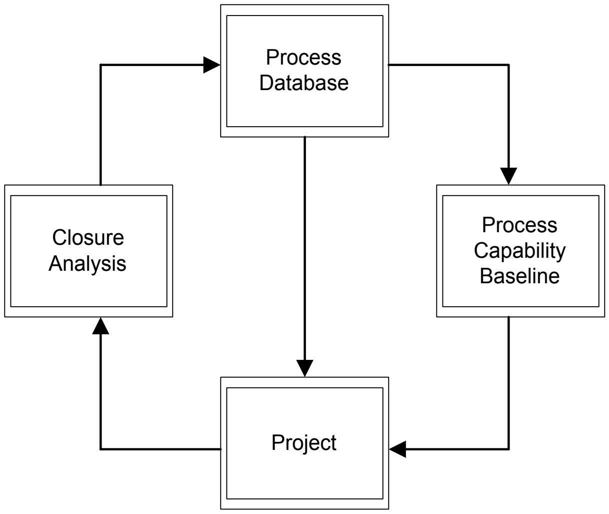

Figure 1: Process database and process capability baseline

The process database captures data on completed projects only. During the execution of a project, data on effort, schedule, defects etc. have to be recorded regularly, if the data discussed above is to be available at project completion. Most organizations, as part of metrics infrastructure, also provide tool support to projects for data entry and analysis. The tools varied from in-house developed tools to commercial tools to simple spreadsheets. Data on effort was generally recorded daily and submitted weekly, while data on defects was recorded after each quality control task. This data is used heavily for project monitoring and control. At the end of the project, the data is analyzed and a summary of data is added to the process database.

Use of Metrics in Planning

The main use of historical data in planning is for effort estimation. Most organizations use the process database directly and use data from similar projects for estimation. However, a few of the organizations also use productivity data in the capability baseline for estimation. (See Figure 1). Organizations generally do not use the "computed" estimated directly and use expert judgement or judgement of the project team to "correct" the computed estimate to arrive at the final estimate. Almost all use a work breakdown structure (WBS), coupled with a bottom-up approach, to arrive at the final estimate. Very few used models like COCOMO [Boe81] or other top-down approaches of determining the overall estimated from the overall size estimate [Bas80]. Some organizations used multiple methods for estimating the effort for a project, and then used the different estimates to arrive at the final estimate.

For estimating the schedule, most did not use the "classical" approach of determining the schedule as a function of effort [Boe81, Bas80]. In all the organizations it was accepted that the schedule may be decided based on business needs of the customer. However, all of the organizations "checked" the requested schedule based on experience, past data, the WBS, and availability of manpower.

Most organizations set some quantitative goals for quality for the project during planning. These quality goals were specified in terms of variation of the actual from the estimated of effort and schedule, final quality in terms of defect density delivered, defect injection rate in the project, etc. To achieve the goal, some intermediate "goals" were also set. These were frequently in terms of defects to be detected at different stages [Jal99]. The estimate of defect levels was for phases, or for a smaller task, like a review or testing. (For the latter, control limits are used, and are discussed later.)

The project quality goals frequently were an improvement over the existing performance levels in the organization, and these improvement goals were usually set in the context of overall process improvement goals of the organization. Whenever the project goals were better than the current performance levels, the project plan had some special plans for achieving the goals, as it was recognized that using the standard processes will result in achieving only the existing performance levels. These project-level process improvement plans mostly focused on improvements in quality control activities and defect prevention.

Besides planning for cost, schedule and quality, risk management was another area where past data was used for developing a plan for a new project. Mostly information about the risks and risk mitigation strategies was used, though some also used the data on probabilities and costs associated with risks.

Use of Metrics for Project Monitoring and Control

When a project is executing, it is important that it is properly monitored. The basic purpose of monitoring is to ensure that the project continues to move along a path that will lead to its successful completion. During project execution, how does a project manager know that the project is moving along a desired path? For this, visibility about the true status of the project is required. As software itself is invisible, visibility in a software project is obtained through observing its effects [Hsi96]. Providing proper visibility is the main purpose of project monitoring.

There are two key aspects to project monitoring. First is to collect information or data about the current state of the project and interpret it to make some judgements about the current state. If the current state is a "desired state", implying that the project is moving along the planned path, then monitoring provides this visibility and assurance that things are indeed working fine. But what if the monitoring data reveals that the state of the project is not "healthy", that is, the project is not progressing as planned? Clearly, then it must be followed by some actions to ensure that the course of the project is "corrected". That is, some control actions are applied to the project. This is the second aspect of project monitoring – applying proper controls to bring the project "back on track". This collecting of data or information to provide the feedback about the current state, and then taking corrective actions, if needed, is a basic paradigm of project management, and is shown in Figure 2 [Jal99]. For project management, this is the main use of software metrics [Bro96].

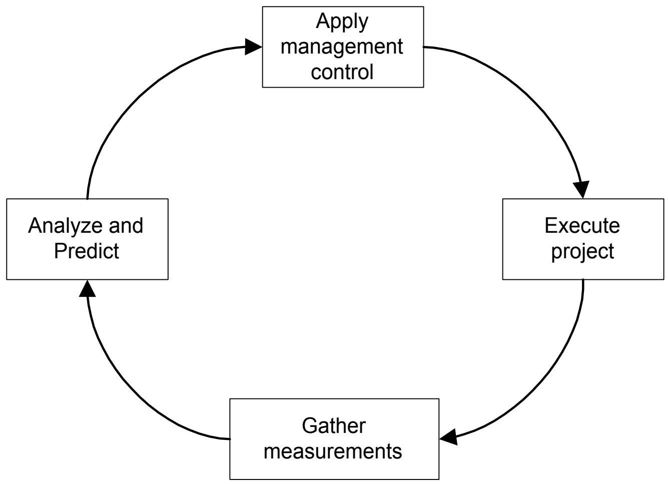

Figure 2. Project monitoring and control cycle

For quantitatively monitoring and controlling a project, two approaches commonly used are (these are over and above the methods that may be used for general status reporting):

- Analysis at milestones
- On-going, event-driven analysis

All the organizations did milestone analysis. Most of them did this analysis with some fixed periodicity (generally monthly), though some of them did this analysis at some project defined milestones. In this analysis, almost all did an actual vs. estimated analysis of effort and schedule. Some also did a causal analysis of defects found so far. The organizations that predicted defects for phases, also did actual vs. estimated analysis for defects. All organizations have set thresholds for acceptable variation in performance from planned. The threshold for effort, for example, varies from 10% to 35%. The thresholds for schedule are generally lower. In most organizations, these thresholds are set based on experience, comfort level, and past performance data. Besides these, in the milestone analysis risks are reanalyzed, cumulative impact of requirement changes analyzed, etc.

Many organizations also applied monitoring and control at event level to provide better control. The organizations that did on-going, event-triggered analysis, generally used statistical process control (SPC) in some form. With SPC, a run chart of some performance parameter is plotted. Based on the past performance data, control limits are established. If a point fall outside the control limits, it is assumed that there are some special reason for it. The case is then analyzed and the special cause found and removed. By doing this the process stabilizes and gives predictable performance (for definitions of SPC, the reader may refer any standard book like [Mont96, Whe92]. For more discussion on use of control charts in software, the reader is referred to [Flo99]).

A sample control chart for unit testing is shown in Figure 3. In this, each point represents the performance of a unit testing (in terms of defects per LOC detected.) Control limits are established, based on past performance, such that if the unit testing process is working "normally", the point will fall within the control limits. If performance of unit testing of a module falls outside the control limits, it is assumed that there is some special cause for the change in

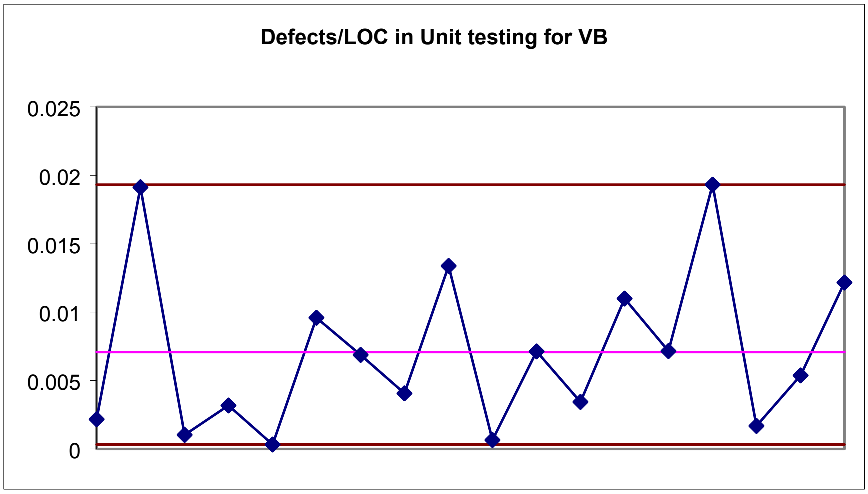

Figure 3. Control chart for unit testing

performance. The analysis may consider effort spent, quality of estimation, quality of code, etc and may recommend actions like re-testing, extra testing later on, revise estimates etc..

SPC is used most frequently for code reviews, though it was also used for unit testing, and sometimes other testing also. Use of XMR charts was most common, though simple x-charts were also used (for definition of these charts and other aspects of SPC, the reader is referred to [Mon96, Whe92, Flo99].) For code reviews, some used organization-wide data to build control charts, while some built control charts used by a project from data from reviews from that project itself (the first few reviews were used for this.)

The control limits were defined as 3-sigma in most organizations, while in some it was defined as the spread in the past performance data or something else. Most organizations used some software to help maintain the control chart. The organizations that built “global” control charts, generally built separate charts for different programming languages (for code reviews) and either one control chart for document reviews or different control charts for reviews of different types of documents. The actions taken by most organizations when a point fell outside the control limits included changing the checklist, improving training, etc. to improve the process, and taking actions like re-review, careful testing in the future, etc. to improve the product.

Use of Metrics for Process analysis and Improvement

With the process capability being computed on regular intervals, trends in process capability over time can easily be analyzed. Most organizations computed the process capability either yearly or half-yearly.

For improving the organization process, most organizations set some organization-wide goal. This goal may be set in terms of improvement in quality and productivity, reducing the variance in performance, reducing the defect injection rate, or improving the review effectiveness. These goals are generally accompanied by some overall strategy. Frequently, each project devises its own plans to meet or beat the organization improvement goals (see discussion above).

Many organizations were able to quantify improvement. One organization that sets very aggressive goals for its quality improvement has seen a 30% reduction per year in its defect injection rate for the last few years. Another organization has seen a reduction of about 50% in delivered defect density as well as defect injection rate in a year. One organization reported a 50% improvement in productivity in 5 years, while another saw its effort overruns go down from 30% to 20% and then further to 10%.

Some organizations were using metrics consistently to evaluate technology and process initiatives for improvement. These organizations had a defined procedure for first implementing a change or new technology in one or more pilot projects, set quantitative improvement goals for this pilot, and then evaluating the pilot against the goals. The evaluation of results from a pilot was used to decide how deployment was to proceed, for example, whether the entire organization should use it, or some type of projects, etc.

Summary

At high maturity organizations, metrics is expected to play a key role. Though metrics data is collected, and even used at level 2 and 3, from level 4 onwards, metrics are expected to play a key role in overall process management as well as in managing the process of a project. The aim of this study is to see how similar are the metrics programs of high maturity organizations, and the nature of the similarities, with the hope that this information can help other organizations in building or improving their own metrics program and in their quest for high maturity.

In a software organization, there are three main uses of metrics data: for project planning, for monitoring and controlling a project, and for overall process management and improvement. To support these uses, some metrics infrastructure is needed. These four areas form the core of this study.

For conducting the study, a few high maturity organizations were selected. The author visited these organizations personally to understand the role and use of metrics. The information collected during these visits was the main source for preparing this report. One of the criteria for selecting the organizations to be included in the study was that their lead-assessors should be different, so that the view of the assessor does not bias the study.

The study found that most of the organizations studied collect similar data that focused on effort, defect, size, and schedule. Most organizations have a process database that maintains metrics data for completed projects. Capability of the process is determined from the past data. For planning, past metrics data is used for effort and schedule estimation, though it has also been used for setting quantitative quality goals. For monitoring a project, all organizations have a regular metrics analysis that focuses on estimated vs. actual of the parameters that have been estimated in the project plan. Many organizations have enhanced it by statistical process control techniques to smaller tasks like reviews, unit testing, etc.

For overall process management, most organizations analyze past data to see the trends in different parameters. Some also use it to set organization goals for improvement. Different types of improvements have been observed by these organizations. Improvements were observed in terms of reduction in delivered defect density, improved productivity, reduction in defect injection rates, reduction in effort overruns, etc.

Acknowledgements. The author is grateful to all the companies (a total of about 8) that participated in the study, and to their representatives who read the initial report and provided valuable feedback.

References

[Art97] L. J. Arther, “Quantum improvements in software system quality”, *Commn. Of the ACM,* 40:6, June 1997, pp. 47-52.

[Bas80] V. R. Basili, *Tutorial on models and metrics for software management and engineering,* IEEE Press, 1980.

[Boe81] B. Boehm, *Software Engineering Economics,* Prentice Hall, 1981.

[But95] K. L. Butler, “The economic benefits of software process improvement”, *Crosstalk,* July 1995, pp. 10-19.

[Bro96] N. Brown, Industrial-strength management strategies, *IEEE Software*, July 1996, pp. 94-103.

[Dia97] M. Diaz and J. Sligo, “How software process improvement helped Motorola”, *IEEE Software,* Vol. 14, no. 5, Sept 1997, pp. 75-81.

[Dio93] R. Dion, “Process improvement and the corporate balance sheet”, *IEEE Software,* July 1993, pp. 28-35.

[Flo99] W. A. Florac, A. D. Carleton, *Measuring the Software Process – Statistical Process Control for Software Process Improvement,* Addison Wesley (SEI Series on Software Engineering), 1999.

[Hal96] T. J. Haley, “Software process improvement at Raytheon”, *IEEE Software,* Nov. 1996, pp. 33-41.

[Her96] J. D. Herbsleb and D.R. Goldenson, “A systematic survey of CMM experience and results”, *18th Int. Conf. On Software Engineering,* Berlin, 1996, pp. 323-330.

[Her97] J. Herbsleb et. al., “Software quality and the capability maturity model”, *Commn. Of the ACM,* 40:6, June 1997, pp. 31-40.

[Hol97] C. Hollenbach, et. al., “Combining quality and software improvement”, *Commn. Of the ACM,* 40:6, June 1997, pp. 41-45.

[Hsi96] P. Hsia, Making software development visible, *IEEE Software*, May 1996, pp 23-26.

[Hum91] W. Humphrey, T. R. Snyder, and R. R. Willis, “Software process improvement at Hughes aircraft”, *IEEE Software*, July 1991, pp. 11-23.

[Jal99] P. Jalote, *CMM in Practice: Processes for Executing Software Projects at Infosys,* Addison Wesley (SEI Series on Software Engineering), 1999.

[Lipke93] W. H. Lipke and K. L. Butler, “Software process improvement: a success story”, *Crosstalk,* Nov. 1993, pp. 29-39.

[Mon96] D. C. Montgomery, *Introduction to Statistical Quality Control*, 3rd Edition, John Wiley & Sons, 1996.

[Pau95] M. Paulk, et. al., “*The Capability Maturity Model – Guidelines for Improving the Software Process*”, Addison Wesley, 1995.

[Pau99] M. C. Paulk, “Practices of high maturity organizations”, *1999 SEPG Conference,* Atlanta, Georgia, March 1999.

[Whe92] D. J. Wheeler, D.S. Chambers, *Understanding Statistical Process Control,* SPC Press, 1992.

[Woh93] H. Wohlwend and S. Rosenbaum, “Software improvements in an international company”, *15th Int. Conf. On Software Engineering,* 1993, pp. 212-219.

[Woh94] H. Wohlwend and S. Rosenbaum, “Schlumberger’s software process improvement program”, *IEEE Tran. On Software Engg.*, 20:11, Nov. 1994, pp. 833-839.