A Tailored Approach to Effective and Efficient Software Process Maturity Improvement

Richard Léveillé
Synopsys Inc.
Mountain View, California, USA
Richard.Leveille@synopsys.com

Abstract
A well-defined and executed software development set of processes is essential to bring to market and to support high-quality software. The EDA (Electronic Design Automation) software industry faces numerous challenges by being technology-driven, requiring fast-paced development, increasing complexity of the software, and the growing size of the code.

This paper offers a practical perspective in defining, implementing and rolling out a method to drive process improvement for software product development. An effective and tailored Software Development Process Assessment method fosters and supports process improvement, aligns the organization on a common set of practices, and ultimately drives predictable, high-quality software releases.

1. Introduction

The EDA (Electronic Design Automation) software industry is an integral part of the value-chain enabling the electronic consumer market. The EDA tools enable designers to create and verify complex integrated circuits and SoC (System-on-Chip) designs from concept to silicon. The EDA users are at the forefront of the technology, constantly stretching the boundaries and finding new ways to use the EDA tools.

The EDA software is technology-driven. Our customers constantly develop new technologies that must rapidly be converted into features in our software tools. The development must meet tight time-to-market requirements while maintaining high standards of quality. The EDA tools are used throughout the lifecycle of the Integrated Circuit design (design, verification, simulation, manufacturing, etc.), requiring the tools to interoperate seamlessly with one another.

From a software development perspective, the development processes must evolve to support rapid development, changes in customer requirements, and predictable and repeatable releases. The software is complex with a high degree of code and functionality dependencies with other EDA tool applications. The software applications are large, ranging from ½ MLOC (millions Lines of Code) to more than 20 MLOCs.

At the company level, the organization must rely on a solid set of development processes and exemplary execution; it is the only way to reliably develop and manage our portfolio of over 70 EDA software tools. The set of development processes not only organizes activities but also directly contributes to the quality of the software.

It is within that context of fast-paced development of high-quality and complex software that a Software Process Improvement program was initiated with the objectives of:

- Drive up process maturity at the product team level
- Promote best practices across product teams
- Improve predictability of functionality and quality
- Increase productivity of the organization
- Instill continuous improvement culture
- Enable globally distributed teams to collaborate efficiently

This article describes a simplified and practical model to continuous process improvement. First, it defines the Software Development Process Model which describes the Desired State for product development.
Second, we describe the Software Process Assessment used in measuring process maturity toward achieving the Desired State. The article provides insights of the results of our program, as well as the key success factors for a practical model for continuous improvement.

2. Bringing focus to Quality-by-Design

The Software Process Improvement program is an integral part of Synopsys’ Quality Management System. The system provides the necessary framework for continuous improvement. The Quality Management System represented in Figure 1, shows product development activities being performed in a closed loop system. The metric system ensures data-driven decisions are made. The continuous improvement component of the system keeps the development processes, product quality, and the Quality Management System itself aligned with the business needs.

![Synopsys Quality Management System](image)

**Figure 1: Synopsys Quality Management System**

The Synopsys’s Quality program is anchored by three vectors:

- **Quality by Design**: Developing high-quality software. Focus on design-in quality method and continuous process improvement based on Best Practices for software development.

- **Quality by Validation**: Detect and fix defects before release. Anchored by test methodologies and code analysis tools.

- **Quality by Defect Management**: Timely resolution of customer reported defects.

The Software Process Improvement program brings the focus on Quality-by-Design within the organization. It enforces continuous progress towards process maturity.

3. Software Development Process Model

Improvement and assessment need a reference point as a basis of comparison and to measure progress. The Software Process Model (SPM) serves that purpose by setting expectations and defining the Desired State to achieve in software development. It is tailored to fulfill our unique business objectives while

**Scope**

The SPM focuses directly at the core of software development. It is defined around four phases of development with a strong bias for predictive development.

<table>
<thead>
<tr>
<th>Development Phase</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Planning</td>
<td>Content, Schedule, Quality</td>
</tr>
<tr>
<td>Specification</td>
<td>Requirements, Functionality</td>
</tr>
<tr>
<td>Design &amp; Implementation</td>
<td>Architecture, Design, Coding</td>
</tr>
<tr>
<td>Test &amp; Validation</td>
<td>From Unit Testing to 1st article testing</td>
</tr>
</tbody>
</table>

**Figure 2: Scope of Software Development Process Model**

Unlike commonly used assessment and audit methodologies, and because assessments are conducted at the individual product level, the scope does not include our corporate-level framework and infrastructure processes for Configuration Management, subcontract management, Quality policy, etc., which are typically addressed in many other formal maturity model assessment programs.

**The Development Reference Model**

The Reference Model is the yardstick against which product teams are compared. The Reference Model is derived from best practices in use within the company and proven within the software industry. This ensures de facto credibility to any recommendations and also provides experience that can be shared across teams. The Reference Model describes the software development practices but maintains some flexibility for the Team to adapt the processes to their size and business-specific challenges.

The Reference Guide is defined and approved by the Engineering Council, a group of all Development vice-presidents of the organization. It sets clear expectations of results and required outcome for each of the four (4) phases of development. The Reference Guide is supported by documented Best Practices, Guidelines, Checklists, templates, and tools. This collection of resources provides teams with a set of proven steps in producing successful EDA software.

**4. Software Process Assessment**

The Software Process Assessment establishes the necessary feedback mechanism for continuous improvement. The assessment is specific to each product team. The process is tailored to scale to the number of products and the improvement cycle for the product teams. The Assessment takes place every 12 months for each of the products. The Figure 3 summarizes the assessment process which encompasses two phases. The Corporate Quality Team is responsible for the overall Program and the execution of the process.
An Assessment Team is responsible for product assessment, bringing an independent and objective view to the Product team. The Assessment Team is composed of representatives from the Corporate Quality Team and one Guest Development vice-president from another product team. The reason for having a Guest vice-president is to enable “cross-pollination”. The Vice-president brings his/her own development experience to evaluate practices, as well as to learn from the Team and possibly discover practices that his/her own team would benefit from. The participation is also key in showing full management commitment to the Program.

**Fact-finding Phase**

Before conducting the Assessment, a series of activities take place during the Fact-Finding phase. This Fact-finding Phase is two-folds:

a) collect feedback from product team members

*Input from Individual Contributors:* The purpose is to obtain feedback directly from developers, QA, Technical writers, program managers, marketing, etc. on their day to day experience in order to assess breadth and depth of practices within the Team. The feedback is obtained using web surveys tailored to the individual functional role.

b) collect in-process data and review development documents

*In-process data and review development documents:* The objective is to perform a detailed review of development documents, schedules, process descriptions, etc. The Team collects In-process data (Schedule: Plan Vs. Actual, number of changes to the release content, Functional specifications, code review findings, Test Plans, quality reports, etc.) that serves to demonstrate existence, use of processes, and results.

This phase helps the Assessment Team to identify less mature areas to better target the assessment. This phase is complete with the release of a Pre-assessment Report to the Product team.

**Assessment Phase**

The next phase is the Assessment Phase. The assessment is a 1 to 2 hour meeting. The meeting participants are the Assessment Team, representatives of various functional groups from the Product Team, and the Engineering Vice-president of the Product Team.
The meeting provides the opportunity for the Product Team to share progress since the last assessment, and to comment on the findings from the pre-assessment. The meeting is an open discussion guided by questions from the Assessment Team.

The Assessment Team uses the Software Development Process Model as a reference. The level of maturity of the development practices is evaluated using the “4E model” for process maturity.

**The “4E Model” of Process Maturity**

The model for Process Maturity reflects the team’s level of proficiency in defining, using, managing, and optimizing processes. The “4E model”, developed internally and summarized in Figure 4, is used to evaluate process maturity: **Existence**, **Execution**, **Effectiveness**, and **Efficiency**. The 4E Model puts emphasis on process results.

- **Existence** designates a state where the Team can identify, describe, and refer to a process the Team recognizes. Key question is: “What is the process?”
- **Execution** is the usage of the process and how well the Team uses it to accomplish the process outcome. The Execution must be part of the Team’s operation and used across the entire team. Key question is: “How well do we execute on the process?”
- **Effectiveness** focuses on the results of the process and the quality of those results. Key question is: “Does using the process produce the expected results?”
- **Efficiency** refers to the approach, timeliness, and resources assigned to execute on the process. The underlying implication is that results are measured and measurements guide improvement. Key question is: “How efficient are we in achieving the results?”

![Figure 4: 4E of Process Maturity](image)

**Final Assessment Report & Action Plan**

The Final report is the main deliverable of the Assessment process. It provides:

- Highlights of the team’s process strengths
- Recommendations on key areas for process improvement
- Score: Process Maturity Index (PMI)

The report addresses in detail all key processes for each of the development Phases identified in the Development Reference Guide. The report enables the Product Team to make fact-based decisions as to what processes are in need of improvements and others that are in good shape.

The report also provides a PMI rating for each development phase. PMI, the Process Maturity Index, is a metric of the Synopsys Metric System used to quantify the level of process maturity, allowing measure of
progress over time. The rating is tied to the 4E Model where greater rating matches the progression from Existence to Efficiency.

Following the assessment, the Team reviews the feedback, determines its own priorities for best ROI (return-on-investment), and establishes an improvement plan. A subsequent incremental assessment will validate progress on targeted areas, as well as confirm sustained proficiency of other practices.

5. Impact of the Process Improvement Program

After many years of having the Software Process Improvement program in place, we do observe a direct impact on process maturity of the organization. First, we notice greater awareness and a more focused mindset around processes. The fact that a discussion around processes takes place creates some dialogue within the organization, forcing the Product team to discuss the way they do things. That simple exercise gets teams out of their comfort zone but helps promote frank discussions to identify how development can be improved.

A continuous improvement culture is taking roots. The Product Teams are taking actions by putting improvement plans in place and actively participating in open discussions. The Teams have shown a focused effort in areas having significant impact on the product and the organization of work.

From a business perspective, the Program meets its objectives of:

- improving development practices
- promoting best practices within the whole organization
- promoting continuous improvement

The gains are not academic; they are measurable, as illustrated below. All teams managed to improve the level of proficiency in development. The Figure 5 illustrates the gain over the past two years which accounts for typically two cycles of assessments.

The realized gain is concentrated in areas having wide impact on the team’s development processes like better Requirement and Functional specifications, improved Test Plans, increased adoption of code reviews, etc. We observe emphasis in all processes supporting better coordination and communication for global team. Most of the improvements have been made in the Planning Phase for higher benefits. Other gains are relative to better definition of requirements and traceability throughout the entire development cycle. The teams also focused on cross-product dependencies, a key element for predictability of releases and also for higher quality of the product.

As an organization, the Program has helped to promote greater exchange of ideas and methods between
the product teams as well as accelerated the rate of adoption of best practices.

6. Key success factors

There are many factors that contributed to the success of the Software Process Improvement Program. Some of the factors are similar to the ones observed in other lightweight assessment methodology [2].

Simple and practical Development Reference Model: The Reference Model is practical and it anchors the discussion while setting clear expectations within the organization. It is kept simple for product teams to relate to and to facilitate quick adoption of best practices.

Sponsorship and Involvement of senior management: The Program is sponsored by the Engineering Council. Engineering vice-presidents directly participate as part of the Assessment Team. This provides legitimacy to the entire program and clearly demonstrates management commitment to continuous improvement.

Members of all functional groups participate in the process: One key element contributing to better accuracy of the assessment is the fact that members from various functional roles within the development team provide input about the processes, their use and effectiveness. The open discussion and wide participation contribute to keep the assessment close to the work and tasks the product team performs.

The Product Team decides on best ROI to maturity: The product team benefits from an independent and impartial view of their processes and state of maturity, but the Team itself decides what to improve. The assessment provides them with facts, strength and weaknesses, allowing them to make decisions in choosing improvements with the best ROI to concentrate on first.

7. What is Next?

The Software Process Improvement Program in itself is built for continuous improvement. Since it is based on best practices, any process innovation that takes place is added to the Reference Model, thereby gradually but continually raising the bar. In addition, now that essential building blocks are in place, more refined processes can be introduced.

Now that progress has been made by all teams to fulfill requirements of Existence and Execution of processes, our next iteration of assessments will be more targeted on Effectiveness and Efficiency of processes; the key elements of higher levels of maturity.

8. Summary

The approach taken in tailoring the Software Development Process Model and the assessment method has greatly contributed at the establishment of a continuous process improvement culture. The Development Reference Model and the associated best practices provide clear expectations to the product teams while aligning the organization on a common set of practices, and ultimately driving high quality software releases. The main impact has been to bring rigor to the development process as well as facilitate the adoption of best practices. The tailored approach allowed us to keep process improvement simple and cost effective, while providing value quickly within the organization.
9. References


10. Acknowledgments

Sincere thanks to Marian Morrison and Pramod Suratkar for their constructive feedback.