A Stitch in Time Saves Nine

Abstract: Organizations often fail to optimize the collective use of engineering tools and setup despite making huge investments. I have mostly experienced that problems exist due to lack of cohesiveness among tools, processes, and development methodology. Before making further investment in a new tool to solve your pain points, I strongly recommend launching an initiative to analyze how different pieces tie-up with each other. In this article, I have tried to share my experience of conducting such a holistic exercise that can culminate in a profitable result.

Article:

With my experience of working as a process consultant, I have found that one of the major challenges faced by the teams is to make the best use of engineering platform. In general, an engineering platform may be described as an integrated suite of tools that facilitate software development lifecycle. I have quite often seen that teams blame existing engineering platform for various troubles related to inadequate productivity, incorrect reporting, inability to link artifacts, lack of visibility, and so on. Sometimes, people expect tools to work for them like a magic wand, which can address their needs and wants. I have also seen people having naive expectations from the tools, e.g., - *'we need a tool that can manage business requirements'*, *'we need a tool that can generate requirement traceability'*, and so on. However, no tool is capable of addressing your needs, unless you clearly define what works best for you. For instance, pre-requisite for implementing a requirement management tool would be a flexible, adaptable, and responsive workflow that simplifies the way requirements are gathered. Similarly, requirement traceability cannot be established unless all software artifacts are linked together in a cohesive manner. Thus, best practices and '*light-weight*' processes are the main driving forces behind successful implementation of any engineering platform. Lack of association between tools and processes can be counterproductive just as computer hardware cannot work without software or firmware. Before setting aside budget for engineering platform, one should make reasonable investment to analyze existing processes in order to identify gray areas that may not be apparent in routine work. Sometimes, such a process assessment is kept on the back burner due to dire need to acquire a new tool, lack of time, or budget.

If you think you are unable to achieve desired results with your existing engineering platform, there is never a wrong time for you to perform a holistic exercise to assess end-to-end processes and development methodology vis-à-vis your engineering platform much before considering new tools. Based on my learning from assessments of distributed teams, I have tried to exemplify problem areas that are out of sync –

- *Inadequate Integration of Tools* – you may not be easily able to track files modified to fix a defect in absence of the integration of defect tracking system with source repository. Similarly, tracing back unstable requirements will not be an easy task if defect tracking and requirement management tools are not integrated to enable the smooth flow of information.

- *Inconsistent Tools and Practices* – this situation is very likely to occur after merger or acquisition of different organizations. Problems crop up when distributed teams use multiple tools for the same task. For instance, code integration problems are very likely to occur if teams in different locations are using different source control tools. On the other hand, simultaneous defect fixes done by the developers from different locations may not be properly updated in source code repository due to inconsistent check-in practices.
- *Tools Unfit for Development Methodologies (i.e., Agile)* – Agile methodologies are characterized with enhanced emphasis on collaboration. Agile practices can perhaps start to subside due to inability of development tools to integrate with collaboration systems, e.g., Wiki. Development tools that do not support real-time collaboration are not the best fit for distributed Agile.
- *Process Overhead due to Complex Workflows* – process workflows primarily drive engineering tools. Simple workflows allow tracking things effectively and thereby reducing process overhead. For instance, too many statuses and transition states in a defect tracking system may interfere with routine work and generation of reports and metrics. Sometimes, I have also seen that unnecessary statuses (i.e., On Hold, etc.) become a convenient bucket to throw tricky issues. Simple workflows are easier for the team to use and thus improve adoption and compliance.

Let's look at systematic steps that can help you to skim through entire development lifecycle in order to gauge suitability of your engineering platform. Key finding from the review will certainly help optimizing engineering platform for best results by enhancing tools, processes, or both.

Identify Blockades

It is essential that processes simplify routine work instead of complicating them. So, it becomes necessary to review existing processes to uncover major pain areas, bottlenecks, and tool inconsistencies. Before planning to perform the review, you should prepare a questionnaire covering key process areas. These are some basic examples –

1. What is the main source of business requirements? How are they captured and which tool is used for this purpose?
2. How do you estimate efforts for a new requirement? Where do you store estimates?
3. How builds, releases, and iterations are planned and tracked? Which tool is used for this purpose? How do you find earned value, planned value, etc.?
4. How do you manage and track requirements (or story), test cases (or storytests), test results, defects? Do you map all artifacts; if yes, how?
5. Is build/ deployment process automated? If yes, which tool is used? How do automated scripts fetch inputs from other systems?

While designing this questionnaire, you should bear in mind that the ultimate goal is to uncover gaps. Passing the buck to other teams, cross-functional groups, and tools is an obvious trend for large teams. Encourage stakeholders from all levels to participate in this

exercise in order to get 360^{o} view of the ground zero. Finally, don't forget to make your own interpretations and invite an outsider with an unbiased viewpoint. You should also build consensus among all stakeholders with the intent to normalize the top priorities. A template shown below may give a good start –

Question	Response – Team A	Response – Team B	Response – Team C	Gap Analysis	Delivery Risk (Critical/ High/ Medium/ Low)	Inference
Requirement and Task Management						
Question 1						
Question 2						
Question N						
Issue Tracking and Release Management						
Question 1						
Question 2						
Question N						
Quality Engineering						
Question 1						
Question 2						
Question N						
Design, Coding, and Configuration Management						
Question 1						
Question 2						
Question N						
Collaboration and Communication						
Question 1						
Question 2						
Question 3						
Question N						

Figure 1.0: Questionnaire Template

Review Gaps versus Top Priorities

Based on review findings, determine your problem areas that need immediate attention. Do whatever is best in your opinion – revamp processes, customize/ re-configure tools, consider new tools, etc. You should evaluate suitable tools if existing tools are not aligned with your requirements. Narrow down your selection after evaluating potential tools.

Create Proof-of-concept using Potential Tools

Before finalizing a tool, you should create a prototype that simulates your existing system in order to analyze feasibility of basic expectations. If you are confused between two tools, you should create prototypes using both. Define your best practice while creating

the prototype of proposed system. Key findings of review will help you to ensure that you are not repeating previous mistakes.

Migrate Data from Existing System

Migration of real data from existing system is an essential activity that you should try while working on proof-of-concept. Successful import of data into the proposed system will give enough confidence for future migration. This will also provide an opportunity to map data entities of existing and proposed systems ahead of time.

Test and Go live

You should extensively test the prototype of proposed system with a selected group of users from the engineering team. It is important that people in this group are well aware with problems faced with the existing system. It will be great if all users perform different development tasks by simulating a mock development cycle in order to determine limitations, if any. In case of a large engineering setup, it becomes important to test the performance by using the proposed system over a period. Decide a date to go live if everything works fine. You are all set to roll out a new system that will help you to get rid of previous problems.

About the author: Mayank Sharma has 13 years of experience in the field of Software Quality Engineering. Presently, he is working as "SQA Manager" in AMI Center of Excellence of Landis+Gyr based in India. In the past, he has worked with various software services and product organizations, i.e., GlobalLogic, Symantec, GrapeCity, Motorola to name a few. I can be reached at mayank.sharma@landisgyr.com or mayanksharma123@hotmail.com.

Disclaimer: Statements and opinion expressed in this article are those of the author. This opinion does not relate to his present or past organizations.