

2008

PACIFIC NW SOFTWARE
QUALITY
CONFERENCE



COLLABORATIVE
QUALITY

OCTOBER 13-15, 2008

CONFERENCE PAPER EXCERPT
FROM THE

CONFERENCE
PROCEEDINGS

Permission to copy, without fee, all or part of this material, except copyrighted material as noted, is granted provided that the copies are not made or distributed for commercial use.

IT Collaboration at Stanford University

Beyond Quality

Claudia Dencker
Administrative Systems, Stanford University
Stanford, CA 94305
650-736-0599

Abstract

In February 2007, the Quality Assurance organization in Administrative Systems (AS), Stanford University was formally underway with two mandates in mind: to set up an independent testing function and to improve the quality of AS delivered software solutions. The challenges that these two mandates attempted to address are as follows:

- Inconsistent (and in some cases, non-existent) processes, tools and procedures across all practice areas within AS
- Lack of transparency of project activities within and outside of AS
- Bad software quality upon delivery

Today significant progress has been made on all three challenges. Silo'd practices are slowly merging into a cohesive whole, projects are more open and transparent with business partners actively participating in project progress, and software quality is improving in key areas.

This paper reports on the progress that the QA organization has achieved since its formation and highlights the unique role that collaboration tools and processes have played in improving business partner satisfaction in AS' delivered solutions at Stanford University. One project will be profiled, the PeopleSoft Student Administration and Human Resources version 9 upgrade project that affects all business offices in support of the university's mission of teaching, learning and research. Tools used are a wiki integrated with an issue tracker, test case management, project dashboards and more.

The more open mode of working with staff across different job disciplines, in different physical locations and in a few cases, across multiple time zones has led to a much higher degree of satisfaction for all team members. Additionally, the university has benefited where individuals are accountable, projects are more disciplined and structured and transparent through the use of an open collaboration set of tools and processes.

Note: Products listed by name in this paper do not imply endorsement. Rather they are listed to provide context for the tools being used by Stanford University, AS Department.

Bios

Ms. Dencker is Director of QA, Administrative Systems, Stanford University. She was formerly Program Coordinator of the newly consolidated program, SEQ (Software Engineering and Quality) at University of California, Santa Cruz Extension and President of Software SETT Corporation, a company that provided QA and software testing solutions to software IT professionals worldwide. She has trained professionals worldwide in software testing and test management as well as led and managed teams on many projects in Silicon Valley. Ms. Dencker is a Certified Software Quality Engineer (CSQE) and graduated from San Jose State University with honors.

Introduction

Bringing a Software Quality organization into a university unfamiliar with such a function has been an enormous challenge. But as most universities today increasingly rely on software solutions to run their “business” of learning, teaching and research, traditional software development practices are folded in. Stanford University is no different. Many of the business functions of student admissions, student records, financial aid, payroll, and more are solely driven not by punch cards and paper documents, but rather by highly sophisticated, web-enabled, self-service software solutions. All the off-the-shelf and custom-built solutions must be specified, developed, tested and supported following solid industry practices.

In 2006, the Administrative Systems (AS) group was reorganized into an independent function, apart from its sister organization, Information Technology Services (ITS). This new organization maintains and supports the following applications:

- PeopleSoft Campus Solutions® (Student Admissions, Human Resources), STARS (online training), time tracking and more.
- Oracle Financials® including various custom bolt-ons such as iOU, CMS (Commitment Management System – a forecasting tool) and more.
- Reporting and EDW (Enterprise Data Warehouse)
- Middleware and Integration Services

As a part of this reorganization a need was identified to establish an independent QA group whose mandate was to:

- Set up an independent testing function
- Improve the quality of AS delivered software solutions

Quality of solution delivery had been a troubling problem in past software deployments, and hence, led to the establishment of a group dedicated to solving this issue. But, as most readers will recognize, software quality is not owned by just one department, but rather by all.

As with any endeavor in which ramp-up and results need to be achieved quickly, looking for solutions that span multiple areas is always a big win. This was accomplished through the use of collaborative tooling.

Collaborative Tooling

One of the goals of the new AS, not just QA, was to bring a greater level of transparency to its efforts. This can be a challenge as many university business offices and/or staff don't understand the inherent dynamic nature and inherent susceptibility of software to instability, unreliability and poor performance. Opening up a process that many don't understand invited criticism, anxiety and intrusive micro-management that added a burden to the AS project teams. But, as in all technology driven endeavors, everyone had to learn and come up to speed as to what can and cannot be done.

The ramp-up for greater transparency was facilitated through the introduction and use of collaborative tools. In support of the QA mandate of improving quality, collaborative web-based tooling became an instrumental solution that spanned a number of QA roadmap goals. The collaborative tools that are discussed in the following sections are as follows:

- Issue Tracking, Jira®
- Wiki Collaboration, Confluence®
- Test Case Management and Tracking, Quality Center®

Issue Tracking

The first tool to be introduced was Jira, an issue tracker from Atlassian Software. During the initial phase of investigation as to bug trackers used throughout AS, practice areas were not in sync. AS teams used spreadsheets, a custom form add-on to the call tracking system and Bugzilla. Jira offered a nice upgrade path from Bugzilla, an excellent step-up solution from

spreadsheets, and a more configurable solution from the custom call tracking form. Furthermore, this product has strong ties to the open source community which meant a strong base for quality (100's of developers, testers, users testing it) and rapid pace of enhancements (due to the pressure from a vocal open source community).

The issue tracker workflow was configured to map to optimum quality practices in support of the Find/Fix process. It was quickly rolled-out as a pilot solution for the Reporting practice which was still working off of spreadsheets. The pilot lasted about 6 months with a production Go Live when the Bugzilla data was migrated in.

To enhance use of the solution and to facilitate quick turnaround of issues during the course of a project, AS/QA opened the tool up to business users who participated closely with AS during the two major test phases:

- SIT (system integration test) which is an internal, independent AS test phase similar to the QA or test phase of commercial endeavors
- UAT (user acceptance test) where the Business Affairs central offices and their distributed users test to ensure fit to their needs.

Issue Tracking Benefits

Consolidating onto one web-enabled issue tracker had the following benefits:

- Consistency in how bugs are identified and tracked through the fix process
- Consistency in practice among a diverse group of users who report in to a wide range of organizations
- Transparency of pre-production issues which engages all participants, excludes no one

Wiki Collaboration

Another tool to be adopted was Confluence, a collaboration wiki from Atlassian Software. This tool had already been used by one of the practice areas in AS, the Middleware and Integration Services group as part of their agile practices. All their technical and test documentation, requirements analysis and design were conducted on the wiki. Confluence was embraced by the QA group to post group-specific information such as project updates, new team member info, group mission and mottos, team minutes, etc. The fact that this tool was from the same company that created the issue tracker was an added plus.

No workflow was necessary; we simply created workspaces and let the dynamics of the group determine what was posted and how it was to be used. The use of the wiki was rolled out to all of AS as of the Jira Go Live in December 2007. Some examples of workspaces are as follows:

- Project-based spaces whereby project teams posted information relevant to the project participants. **Note:** The use of a wiki on one of the largest projects in AS is described later in this paper.
- Group-based spaces such as the AS/QA space whereby information was posted relevant to the group members
- Personal pages in workspaces to maintain a log of events

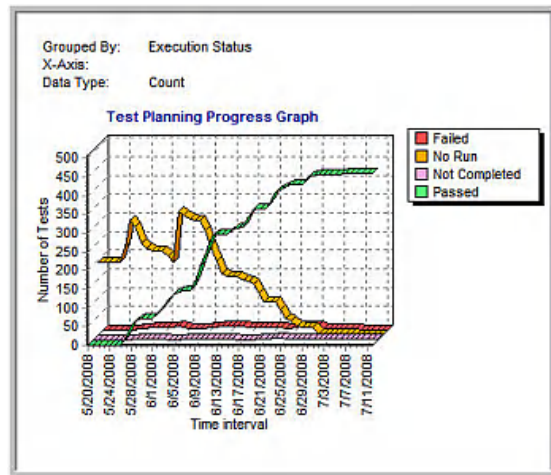
Wiki Collaboration Benefits

Consolidating onto one web-enabled wiki had the following benefits:

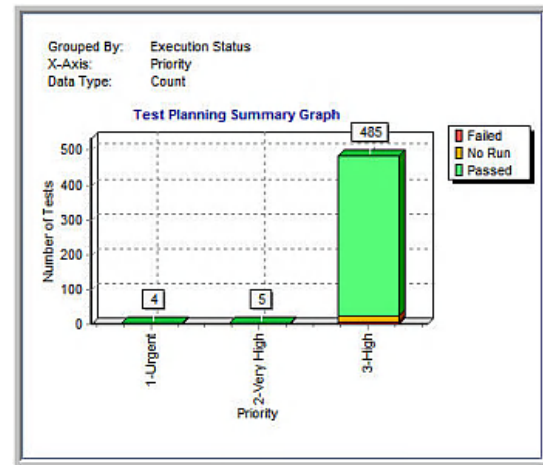
- Relevance of information (assuming information was maintained!)
- Enhanced ownership (team/group members had specific directives to update pages or sections)
- Consistent knowledge about a project or group
- Transparency of effort which was in sharp contrast to the business as usual, opaque/black hole of the past

Test Case Management and Tracking

The third tool was Quality Center from HP for test case creation, management and tracking of test results. This tool isn't typically thought of as a collaboration tool as it was primarily used as a tracking tool from which to draw test case execution metrics. Nonetheless, information was posted in the wiki on a regular basis and as such was folded into the broader collaborative initiative started with Jira and Confluence. Two projects used this tool to upload their test cases, maintain the cases, and to log their test results. The reporting and query analysis is outstanding and hence made for a great addition to the project information that was posted to the wiki.



Progress graph showing planned tests that have passed, failed, not completed, not run.



Planned tests sorted by priority (urgent, very high and high priority) and their status of passed, failed, etc. Medium and low tests not shown.

Test Case Management and Tracking Benefits

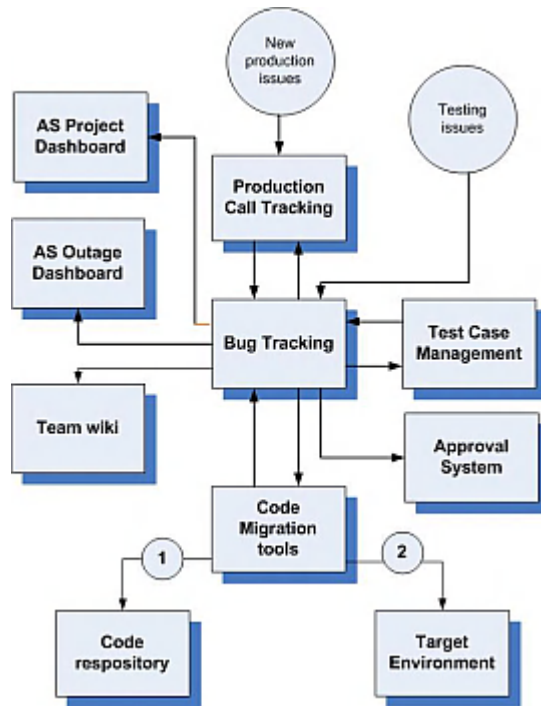
A key benefit from this tool is the accuracy of test execution progress. On many projects estimating progress in an objective manner is rarely done. Rather project team members invariably estimate based on "gut feel" as to their percent complete. These estimates tend to favor the optimistic and are typically wrong. Quality Center removes the guesswork from reporting testing progress subjectively, and maps progress to very concrete, objective criteria for a predictable, solid report to project and executive management.

Secondary benefits are as follows:

- Consistency in how tests are identified and tracked through the test execution process among a diverse group of users who report in to a wide range of organizations
- Enhanced reuse of tests
- Ease of creating test execution runs
- Transparency of test results

Tool Integration Roadmap

The three tools mentioned above are part of a larger roadmap of integrated tooling that will provide an holistic approach to the tracking and management needs of AS. The roadmap is comprised of off-the-shelf, vendor products and home-grown solutions with the intent of more effectively managing the AS systems and communicating changes. Today, four out of the nine applications are integrated.



Call Tracking. New production issues are initially logged and tracked in a Call Tracking system.

Bug tracking. 1) Issues that require technical work or technical assistance are automatically logged into the issue tracker. 2) New issues from various testing efforts are created and tracked in the issue tracker. This includes change requests, bugs, enhancements, tasks and supports a CR, find/fix, migration/approval process.

Team Wiki. Project, group and miscellaneous information is logged and published to all registered users. Issues are automatically published to the wiki. Users must be authenticated to view information.

Dashboards – Public websites. Executive view of AS projects and AS system outage. Details are found in the wiki, the issue tracker, team time tracker or MS-Project® which has a direct feed into the AS Project Dashboard.

Test Case Management – Test repository of test cases and test results. Test cases and test runs are created and run in support of all pre-production test efforts. Reports are published on the wiki project pages.

Approval system. Required approvals are sent via e-mail to university staff not in the issue tracker. Most other approvals for production migrations are handled directly in the issue tracker.

Code Migration tools (deployment manager tools) – Code fix migration information to any environment updated on issue tracker tickets for full end-to-end information about issues.

Code repository – Code repository that is referenced and tracked in deployment manager tools during builds and code migrations to desired target environment.

Tools not on the roadmap but that are used in AS and that foster collaboration are as follows:

- Shared calendar
- Instant messenger
- Online web conferencing
- LCD projectors that project one's laptop image
- Laptops for all staff and mobile devices for selected staff in support of the Stanford Work Anywhere initiative

PeopleSoft v9 Upgrade

So what do all these tools have to do with a real project? How are the delivery practices improved when a large cross-section of the university uses the same open tools? Well, a lot of great things happen. The reach is enormous and the results fantastic.

In 2003 Stanford University upgraded its PeopleSoft implementation from version 7.2 to 8.0. This project resulted in an enormous effort that involved many departments on campus. The primary vehicle for communicating among members was e-mails. Working spreadsheets, announcements and key information would be sent to members on e-mail distribution lists. As the project progressed, staying on top of key communications became more and more challenging as spreadsheets would be updated by multiple people and routed via e-mail, individuals would start silo'd discussions in e-mail inadvertently omitting others, and common, definitive posting of project information was not maintained.

In 2008 this changed as collaborative, server-based communications was instituted by AS/QA during the UAT (User Acceptance Test) phase of the project. Now team members (13 in number) would update wiki pages knowing with confidence that they were working with the most recent version of a page. All issues were tracked in Jira and test results tracked in Quality Center with stats posted weekly. The project dashboard represented accurate information as to overall project progress because the dashboard was populated directly from the time tracking system. Code fix migrations to the various pre-production environments were easily visible on the issue tracker tickets due to the code deployment integration. Essentially the goal was to make all project information and team involvement as open and transparent as possible and to introduce objective criteria by which to measure progress with the net result of building trust and a higher confidence level than on the earlier release. By and large this goal was achieved.

This upgrade was an enormous undertaking before it even reached the UAT phase. The project included the following pre-UAT activities:

- The services of an offshore IT vendor were engaged which included retrofitting of the application's 970 online and 380 batch customizations and conducting SIT (System Integration Testing) on Stanford systems. Over 25 engineers in three locations in India were involved in this effort along with an onshore staff of 5.
- The vendor's performance and quality of deliverables were audited by QA through careful analysis of breadth of testing and depth of coverage which resulted in an increase by 45 % more testing to close the identified gaps (for a total of 1,696 tests executed) and finding 1/3 of the missing tech docs.
- The AS analysts conducted ad hoc testing to complement the strategy employed by the vendor and logged new issues as well.
- Vendor and AS technical staff fixed and retested all logged issues.
- Over 10 environments were created throughout the project to support development, code control and testing.

During UAT, the following key activities took place:

- All collaborative tooling was spearheaded by the newly formed QA group with initial workspace and pages being set up and maintained by QA.
- All planning was done on the wiki and included the following information:
 1. Test team member profiles
 2. FAQ's for common questions raised in the beginning. Not maintained much once effort was underway.
 3. Weekly meeting notes and action items
 4. Team status and metrics (bug and test case metrics)
 5. Calendar (time-off plans, test cycle definitions, end-game plans, vendor contacts, DB links)
 6. Special test activities in support of parallel testing and Full System Integration Testing (FSIT) with FSIT ultimately being moved into Test Director

7. Test summaries

- Team members met once a week in person or dialed in if they weren't on campus. The Stanford business offices and groups that were involved were as follows:
 1. Admissions, which included undergraduate admissions as well as graduate school admissions for business (GSB) and law.
 2. Campus Community, comprising representatives from various offices
 3. Financial Aid, which included the central financial aid office as well as the GSB and Law School financial aid offices.
 4. Graduate Financial Support (GFS)
 5. HR, Benefits, and Faculty Affairs
 6. Payroll
 7. Registrar's Office, including undergraduate/graduate student records as well as the GSB and Law School
 8. Student Financial Services
 9. Oracle Financials (Controller's Office)
 10. ReportMart1 (RM1) reporting

The following organizations not involved in this upgrade because their systems are separate from the university.

1. Stanford Hospital, Medical Centers
 2. SLAC (Stanford Linear Accelerator)
 3. Stanford Management Group
- Test phases entailed the following:
 1. Smoke Testing. This testing was done by most offices at the beginning of UAT to ensure that the v9 functionality, customizations and environment were stable enough for further testing.
 2. Business Testing. This testing was completed by all Business Affairs offices to ensure that Stanford job objectives could be accomplished.
 3. Parallel Testing. Two payroll cycles were formally conducted (mid-month, month-end) to ensure that transactions (such as payroll, billing) in the production system matched the test system or were within acceptable limits in variance. Additional payroll testing was completed during the normal course of business testing.
 4. End-to-End Testing (FSIT). This testing was conducted at the end of UAT, involved all business units, both within Stanford and across vendors, working together to ensure that:
 1. Full system or life cycle processes (manual and/or automated, Registry, vendors, etc.) functioned as expected.
 2. All dependencies were accounted for and correct.
 3. The upgrade has not adversely affected the proper passing of data, events, files, etc.
 5. Performance Testing. This testing took place on the new production boxes to ensure that production performance was acceptable.
 6. Regression Testing. This testing was done throughout UAT to ensure that bugs were fixed correctly and that prior tests continued to provide the same results as before.
 7. Mini-FSIT. This testing was added to provide a higher level of assurance that the system truly was ready for Go Live. It was a repetition of some of the FSIT tests.
 - Approximately 2,400 tests were run with test results logged in Quality Center.
 - Approximately 1,000 issues were logged in the issue tracker with bug fixes tested by AS and QA analysts before being deployed to the primary UAT environment for final verification by the UAT testers.

- Team members were required to update their status on Confluence and in Test Director by 5 pm the day before the weekly meeting.
- QA Lead pulled defect and test case reports and posted in the wiki by 10 pm the same day as the leads.
- Project manager consolidated information into executive reports which were handed out at the weekly governing group meetings and posted on the Project Dashboard.

Team members who had participated in the upgrade years earlier repeated numerous times how much more effective and better the project was run this time around. Everyone appreciated the inclusiveness and openness that the collaborative tools brought to the team's effort. While the collaborative tools and the transparency of the effort were not perfect, it was a huge step in the right direction.

Adoption Challenges

Bringing new tools and processes to Administrative Systems was at times challenging for the following reasons.

Timesink

Change is not always appreciated as it involves an added burden to an already packed plate of work. The value-add of new tools must be compelling in order to break through the resistance that invariably occurs. This was fairly easy to achieve by using an open issue tracker. But, moving from e-mail to a wiki was slower as updating pages and maintaining information on the wiki was more burdensome. E-mail and IM are quick and easy (we're all used to this mode of communication now) but using a wiki requires a "rethink." The wiki has to be logged into, the user has to click on 'Edit', save their work periodically, clean-up the messed-up format of the page and then call it a day. Due to the current state of wiki functionality, it is not a perfect substitute for e-mail though there is a huge momentum by many wiki vendors to beef up the features. Regardless of the burden of wiki content maintenance (i.e., server-based communications), the effort is hugely worth it. Upon project completion, all team members would chant "Put it on Confluence!" when Go Live planning was underway. The benefits of open communications far out-weighed the added time (only in minutes) it took to bring the information to the wiki.

Open versus closed

In silo'd communications information can be more easily hidden or more easily taken out of context. In open communications this becomes much more difficult. If certain team members want to hide progress or project problems, it is considerably easier in the pre-collaborative mode. While it is possible in a collaborative world (just don't bother to track, publish or report on certain aspects of a project), the spirit of openness as represented by the collaborative tooling represents an opportunity for all team members to come together, educate others and truthfully represent what they are working on. This engages team members and management in a proactive manner and ensures that proper management support is brought to bear throughout the life of the project.

Timidity

One of the key hallmarks of wiki's is resistance to editing pages for fear of messing up (for all to see), losing key information (and not knowing how to pull lost information back) or stepping over someone else's edits (with no change marking present). Many people are more comfortable leaving someone else's work to stand on its own.¹ Most wiki's have minimized this by tracking all page changes and allowing users to reinstitute a prior page. To work through the initial timidity of using a new tool many users had to be explicitly directed to what page and where on the page to update. After a few times, everyone started to feel more comfortable and took ownership of their pages or sections of pages. All pages were by and large open to the team with certain critical pages having edit restrictions placed upon them.

Inability to turn off

In a collaborative world, the presence and availability of team members is more visible. We used IM extensively, received nightly news as to updated pages in the wiki over the prior 24 hours, could follow how an issue churned through the find/fix process by reviewing the issue history (who assigned an issue to whom and why), could get onto each other's calendar through shared calendaring and more. This on-going presence of the team 24/7 meant that it was at times difficult for team members to tune out, turn off, and recharge.

Conclusion

The AS/QA initiative of bringing open, collaborative tooling to projects had a major impact in building trust and confidence among all team members. While impacts to software quality are many (rigorous and inclusive review of requirements by all team members including the ultimate users of the solution, SDLC best practices, standard tools and supporting processes, independent testing, and much more), software quality is owned by all and supported through the use of collaborative tooling.

In fact, the experience by team members fit well to how Wikipedia defines collaboration³:

Collaboration is a recursive process where two or more people work together toward an intersection of common goals — for example, an intellectual endeavor that is creative in nature—by sharing knowledge, learning and building consensus.

Recently I came across a slightly better definition of collaboration by Evan Rosen, [The Culture of Collaboration](#).

A working together to create value while sharing virtual or physical space.

The goal of openness, transparency, ownership and consensus among team members from many departments across campus was achieved on the PeopleSoft v9 project during the UAT phase and other AS projects through the use of these integrated collaborative tools.

Using cool new tools is always fun but these new tools alter the way we communicate in a fundamentally new way. Rather than to initiate the communication from our desktop (e-mail) in a client-centered view of the world, these tools allow us to communicate in a server-centered manner with an immediacy not realized to-date. We no longer push information out and wait for others to respond on their time table, rather we pull information in whenever, wherever and however we want it²; information that is open to anyone who cares to look. This engages team members more fully as they are not only passive viewers of the information, but in fact, active originators as well.

Bibliography

Books

Collaboration 2.0, Technology and Best Practices for Successful Collaboration in a Web 2.0 World, David Coleman and Steward Levine, January 2008. Published by Happy About.

² *The Culture of Collaboration, Maximizing Time, Talent and Tools to Create Value in the Global Economy*, Evan Rosen, 2007, Red Ape Publishing.

Unleashing Web 2.0 From Concepts to Creativity, Gottfried Vossen, Stephan Hagemann, 2007. Morgan Kaufmann Publishers.

Internet Links

¹ <http://metamedia.stanford.edu/projects/traumwerk>

<http://c2.com/cgi/wiki?WikiDesignPrinciples>

³ <http://en.wikipedia.org/wiki/Collaboration>

http://en.wikipedia.org/wiki/Collaborative_software

Product Information

PeopleSoft Enterprise Campus Solutions® -

http://www.oracle.com/applications/peoplesoft/campus_solutions/ent/index.html

Oracle Financials® - <http://www.oracle.com/applications/financials/intro.html>

Jira® - www.atlassian.com

Confluence® - www.atlassian.com

Quality Center® -

https://h10078.www1.hp.com/cda/hpms/display/main/hpms_content.jsp?zn=bto&cp=1-11-127-24_4000_100

Bugzilla - <http://www.bugzilla.org/>

MS-Project® - <http://office.microsoft.com/en-us/project/default.aspx>