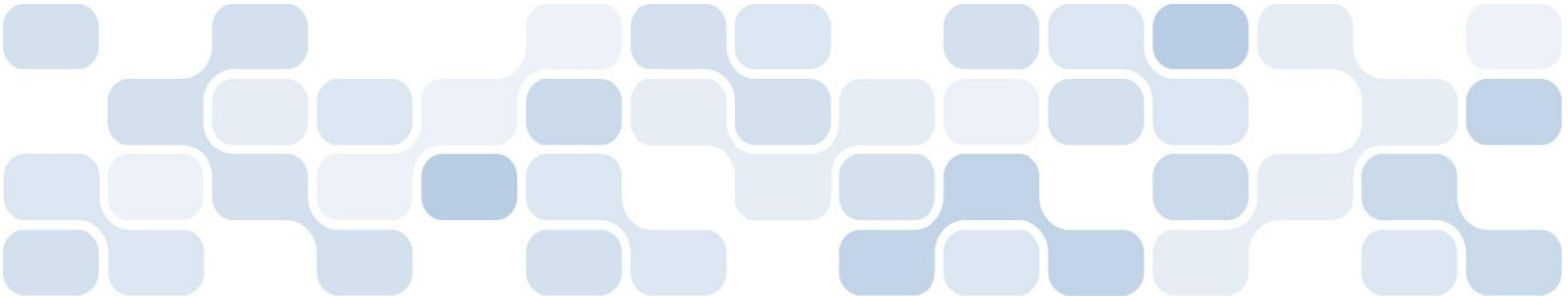
Microsoft

Communicate and Collaborate with Visual Studio Team System 2008

White Paper

May 2008

For the latest information, please see www.microsoft.com/teamsystem



This is a preliminary document and may be changed substantially prior to final commercial release of the software described herein.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This white paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2008 Microsoft Corporation. All rights reserved.

Microsoft, Visual Studio, Excel, SharePoint, SQL Server and Project are trademarks of the Microsoft group of companies

All other trademarks are property of their respective owners.

CONTENTS

INTRODUCTION

Do you remember playing the game “Operator”? You say something to the second person who says something to the third person and on and on until the message comes back to you. The message is never the one that you originally said to the second person. Somewhere along the way someone heard what you said differently or relayed it differently. Software development is a lot like that.

Projects fail (or are challenged) at a very high rate. The reasons for these failures can be linked to many things—poor requirements, poor quality, scope creep, usability issues and other reasons. The underlying cause of most of these issues is that teams don’t communicate well and they don’t have the same understanding of issues which makes it hard for them to work well with each other. In almost every case these are issues that crop up between the development team and the customers.

Whether you are a project manager, architect, developer, end user, tester or another interested stakeholder, Microsoft® Visual Studio® Team System 2008 helps improve the job of developing software. This is done through a free and open flow of information between everyone involved on a development team.

COMMUNICATION CHALLENGES

Development Process

The process of developing a system starts with a business owner identifying the high-level requirements. Then a subject matter expert talks to an analyst who identifies the detailed requirements and documents them. The subject matter expert may or may not be someone who is going to use the system. An architect designs the system based on technology, user requirements and non-functional requirements and provides the architecture to the developers. The analyst creates a functional specification (a translation of the requirement into something a developer can implement) and hands it to the developer. The developer works on the requirements. If the developer has a problem, they talk to the analyst who talks to the subject matter expert to get an answer. When a developer is done, the code is handed off to a tester for testing. After the release the application is handed off to Operations to manage it. Everyone communicates status to the project manager and the project manager reports up through the management chain. Figure 1 shows a simple example of the team communication.

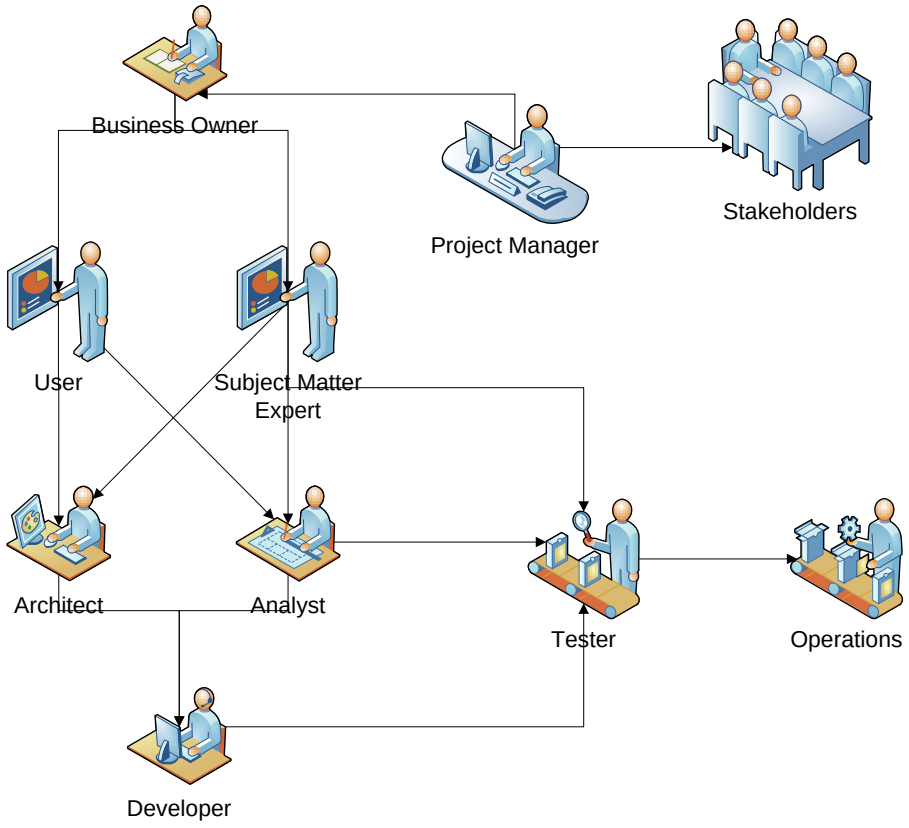


Figure 1 – The development team

As anyone who has worked on a development team can attest, information can become inaccurate or stale by the time it reaches the intended

recipient—if it reaches that person at all. Trying to communicate status becomes a full time job in itself because so many people are working on different things and people who want to track specific issues have a difficult time of it. That's where Team System steps in and excels.

Wasted Time

Not only does this entire process take a while, but people waste large amounts of time trying to effectively communication information. Take the Project Manager—every week the PM needs status from the developers on what they were working on, how long they spent working on it (or how much work is left to do), if they are done with it, and if there is anything blocking them. Developers don't usually volunteer this information—the PM often has to track them down or worse, create a Microsoft Excel® spreadsheet for each of them, e-mail it, and wait for a response.

Or imagine the developer who starts working on a requirement only to find out that the requirement has changed but nobody told the developer! This happens frequently. The developer must fix or change work causing a large portion of the project timeline to alter because someone didn't talk to someone else.

How about a manager who wants status on the project right now? They track down the PM who has to then track down everyone else for an impromptu status which can take a bit of time depending on the team. This wastes the PM's time plus all of the developers have to stop work to figure out their status. And don't even get started on weekly status meetings. These things can take an hour of preparation per developer plus a meeting that lasts an hour. So on an eight-person team that is eight hours of prep time plus eight hours in the meeting (8 people times 1 hour) for a total of 16 hours lost per week in status meetings!

VISUAL STUDIO TEAM SYSTEM 2008 TEAM FOUNDATION SERVER

Core Tenants

You cannot have successful software development without effective communication between stakeholders and development team members. Microsoft created Team System to address many of these communication gaps in the way that many of the stakeholders in a software development project work—Microsoft built it from the ground up for the whole team, not just the developers. This means that Team System doesn't impose itself on you, it works with you.

Microsoft designed Team System to solve a number of problems with the software development process. These include version control, quality and integration issues, and a central location to store information. Solving these individual problems is meaningless without the ability to effectively collaborate and communicate. Through a Microsoft® SharePoint® site for collaboration and sharing, Team System Web Access for easy access to detailed Visual Studio Team System 2008 Team Foundation Server information, the workflow provided by Team Foundation Server, and the reporting provided by Microsoft SQL Server™ Reporting Services, Team System meets and exceeds the goal of open and transparent communication.

Accessing Information

One of the difficulties with other project management tools is being able to access information. Some tools store quite a bit of information but there's no way to get that information easily—it requires a great deal of custom work. Out of the box, Team Foundation Server exposes information through a number of mechanisms to enable team members and other interested parties to access the data they want when they want it. Developers will primarily use Team Explorer, which is an add-in to the Visual Studio IDE. Through Team Explorer they can create and query work items, add, edit, and view documents stored in SharePoint, and view reports from SQL Server Reporting Services.

Project managers can use the SharePoint site directly, which contains documents and reports or they can use tools that they are comfortable with such as Microsoft Excel and Microsoft Project®. With Excel and Project, the project manager can add and query work items, view the status of work items, and create a work breakdown structure based on the work items in Team Foundation Server. Additionally, they can update data bi-directionally between Team Foundation Server and Excel or Project. Imagine now that instead of one project manager being assigned to one project, that project manager is assigned to three projects! Because of the efficiency gains you do not necessarily need a one-to-one relationship between project managers and projects.

Analysts, users, and other stakeholders will likely get their information from two places—SharePoint and/or Team System Web Access. The Team

System Web Access Web site is designed to mimic the functionality of Team Explorer in the browser. This means that users can track issues that are important to them and get an up-to-date status without having to write e-mails and wait for a response from the project manager or developer.

In addition, you can use the capabilities of Team Foundation Server, SharePoint, and SQL Server Reporting Services to get automated updates when documents and work items change. You can have Team System run reports for you on a scheduled basis or when data changes! Team Foundation Server also supports "alerts," which send an e-mail to a user (or Web service) with information on changes to work items, builds, and other information. The alerts can be filtered to send an update when specific events occur. For example, you can receive an e-mail whenever someone assigns you a work item or the testers can be notified when a work item is ready for testing. These help you quickly and efficiently communicate important status to the people who need it. With these tools in place, accessing, sharing, and updating data is easier than it ever has been and helps facilitate the flow of information. And if you really want, and none of the above methods works for you, you can write your application to retrieve data on the fully accessible client API, Web service layer or use one of the many third-party tools on the market.

Work Items

At the heart of all communication within Team System is the work item. Work items contain information which may be related to other information. The other information can take the form of code, linked or attached documents, Web pages, notes or other work items. Work items are also associated with states—that is, they can move from one status to another (along a defined path) which enables you to pass work from one person or group to another (see Figure 3 for an example). Figure 2 shows a list of work items and a single work item selected from the list. Work items can be anything and largely depend on the Process Template you select as the basis for your methodology. Some of the default types of work items available include: Requirement, Task, Bug, Change Request, Risk, and others. You can also use the Process Template Editor to create custom work item types.

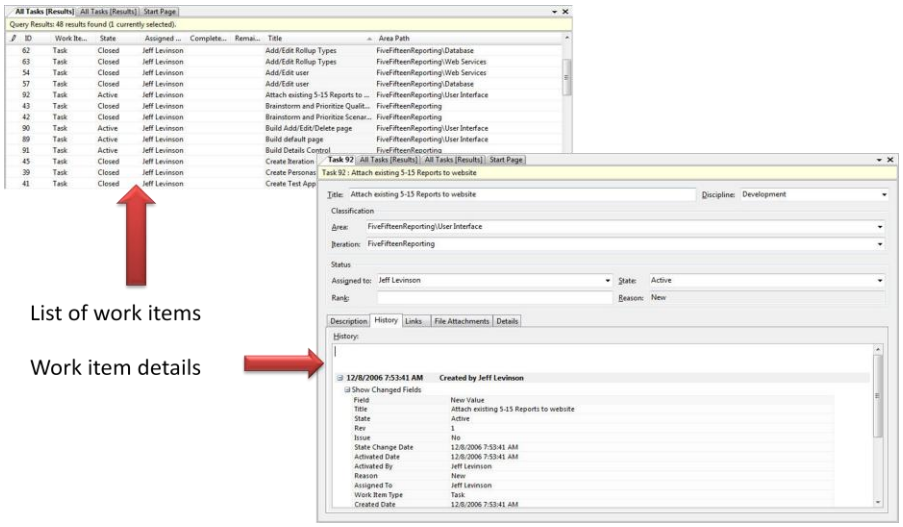


Figure 2 – Work items

The status of the work item supports the communication process by identifying where in the workflow process the work item is. Figure 3 shows the Task states and transitions between states. States are displayed in the blue ovals and the allowable transitions are noted by the red arrows. The text indicates the reasons a transition can occur between two states.

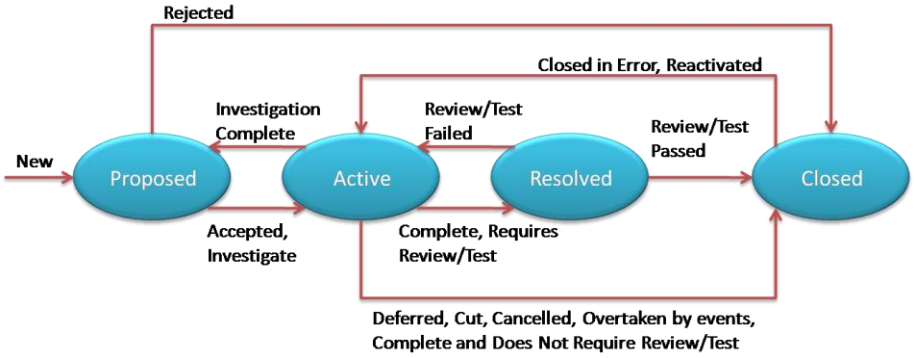


Figure 3 – Task state and transition reasons

Through this fairly simple process, anyone can keep track of a task (the states and transitions are slightly different depending on the work item type but all follow the same type of pattern). At any point, a team member can enter notes against the task and add the detailed reasons. Each work item maintains a complete history so every change against the work item is tracked. This information is used in the reports that are included with Team System or you can create custom reports to query this information. And work items can be reassigned to different users to help move the process along.

Querying for Status

Work items are great for conveying information, but how does this translate into providing, say, status to the project manager? The project manager can

use Excel to query the Team Foundation Server Data Warehouse through a simple-to-use interface to retrieve ad-hoc information about the project status. For instance, they can find out everything that everyone has worked on during a given week. Along with that, the project manager can see the status of every work item (requirement, task, bug, etc.) without the developers having to provide any additional information!

For assigning work, the process works the same way. The project manager assigns work items to the developers and developers can run the My Work Items query to figure out what they need to work on and what priority everything has!

Stale Information

As mentioned before, you can link work items to anything—including hyperlinks that point to documents on a SharePoint site, Web site or server share. With this functionality, users and analysts can work on documentation and store it on the SharePoint site so developers are always accessing the latest version. And because SharePoint stores document history, you can view changes to the documents if necessary.

SharePoint and Web Access

Much of communication is about people doing what is natural and easy to them. If users have to change their habits too much to accomplish a goal, they aren't likely to do it. For this reason, Microsoft integrated SharePoint with Team System. From directly within Team Explorer, developers can add, view, and edit documents on the team SharePoint site (which is created for every new Team Project). This also leverages the built-in functionality in SharePoint, which enables people to collaborate on documents. Multiple users can look at and edit the same document at the same time. Users can ask questions and answer those questions in real time and make the appropriate changes to the document. Everyone will be working with the same version and will quite literally be on the "same page".

RELATIONSHIPS

Communication is at the heart of relationships—with good communication you can have good relationships, with poor communication there is no hope at all. Team System helps to open up lines of communication by facilitating the flow of information between different teams. Developers and testers frequently have difficulty communicating because they don't provide each other helpful information. Developers and users have a "lost in translation" issue because the information users provide gets filtered so much before it reaches developers. Developers and DBAs don't talk until it's too late in the process and then serious disagreements occur. All of these are detrimental to a good working team and Team System can help solve all of these issues.

Developers and Testers

It seems that there is a natural dislike between developers and testers. Testers break the developers' code and tell the developers they broke it but often cannot provide accurate information on what broke and how it broke. The developers then can't reproduce the problem and get frustrated trying to fix whatever broke.

A typical release cycle may go something like this: Developers send the testers the completed code and say "It works, we're done." Testers test the code and, naturally, find bugs. The testers then send this code back to the developers for a fix and the process repeats itself. Today this is accomplished with e-mail, network shares, version control, meetings and telephone calls. The information provided by testers or developers is usually incomplete, which leads to even more frustration. What's lacking is a simple mechanism to communicate the nature of the bug, where it is and which test (and test step) caused the code to fail.

Team Foundation Server provides that tracking and communication mechanism. Because developers and testers can associate test suites with a requirement, each group can look at the other's tests. When a test fails, the test results can be attached to a work item and in one step the developers know which requirement the test is associated with, which test failed, and they can see the test results (including the stack trace). No phone calls, no meetings, no e-mail, and no passing files via shares.

And this process works both ways if developers use unit testing.

Testers can view the suite of tests that the developers created and figure out why they missed the problem (either didn't create a test or the developers' test failed). Either way, this simple mechanism of using work items as the communication channel provides much needed information and improves the working relationships between developers and testers!

Users and Developers

Users and developers may also disagree. Consider what happens when a user reports a bug. They expect to see action taken on it but depending on

the bug, project teams can be slow to respond to user needs. Some project teams don't provide feedback of any kind unless specifically asked. One new feature of Team System 2008 is the ability for someone without a Client Access License (CAL) to enter a work item and view or edit that work item. This means that anyone can submit a bug or a change request (through Team System Web Access) and watch its status as it moves through the system or they can discover if the bug has been deferred. By subscribing to events – which notify users by e-mail of status changes - Team System will automatically keep you informed of the current status. This enables users to keep up to date on the progress of the development team and opens up Team System to an entire class of people who previously weren't licensed to use Team System.

Database Administrators and Developers

As with users and testers, there is a typical disconnect between the Database Administrators (DBAs) and developers. The arguments usually occur because developers write poor SQL code, send their code to the DBAs in an e-mail, and ask them to implement it. The DBAs only record of changes is in the e-mailed scripts. If multiple developers are working on a project then the DBA usually has to collate the scripts sent to them in order to get them executed correctly. This leads to numerous problems including an almost automatic failing of Sarbanes-Oxley and other regulatory compliance measures. In the case of Visual Studio Team System 2008 Database Edition plus the use of work items eliminates these problems altogether. Team System's version control features help to manage and understand change—and Team System helps teams communicate change, especially with the Database Edition. Now developers and DBAs can work together to understand the impact of changes and generate scripts without having to send e-mails back and forth or having the DBA play detective to figure out the changes the developer really wanted.

CONCLUSION

Visual Studio Team System 2008 brings many desired changes and positive outcomes to the development process. This paper discussed the effect of Team System's tools on enhancing (or indeed, enabling) communication between team members to remove inefficiencies that exist within current environments today. As you examine the range of project management tools available to you today, either as a member of the development teams or a manager, you need to ask yourself if the tools you are considering are integrated and easy to use. And if they are, do they provide sufficient benefits. No other tool on the market provides for this level of integration—both in the tool and for the teams using the tool. Team System brings teams together through open communication and easy collaboration between team members, which includes people in the process, rather than excluding them.

ABOUT THE AUTHOR

Jeff Levinson is the Application Lifecycle Management Practice Lead for Northwest Cadence which specializes in Team System and process improvement. You can reach Jeff at Jeff.Levinson@nwcadence.com.

This white paper was developed in partnership with A23 Consulting.