SVN vs. Team Foundation Server [closed]

Asked 16 years, 4 months ago Modified 11 years, 8 months ago

Viewed 56k times

77

A few months back my team switched our source control over to Apache Subversion from Visual SourceSafe, and we haven't been happier.

Recently I've been looking at Team Foundation Server, and at least on the surface, it seems very impressive. There is some great integration with Visual Studio, and lots of great tools for DBAs, testers, project managers, etc.

The most obvious difference between these two products is price. It's hard to beat Apache Subversion (free). Team Foundation Server is quite expensive, so the extra features would really have to kick Subversion in the pants.

- Does anyone have practical experience with both?
- How do they compare?
- Is Team Foundation Server actually worth the expense?

svn tfs

Share

Improve this question

Follow

edited Apr 23, 2013 at 19:52

BryanH

6,042 ● 3 ● 36 ● 47

asked Aug 7, 2008 at 0:43

Brandon Wood

5,337 ● 4 ● 39 ● 31

26 Answers

Sorted by: Highest score (default)

88

Here are the biggest differences between the two for me, and I've used both:

1) **TFS is rather tightly coupled to the "Visual Studio way" of doing development.** That's not to say that TFS is tightly coupled to the VS IDE, it means that TFS struggles to keep the familiar "check in"/"check out" paradigm of Visual SourceSafe, even when it really isn't an appropriate model anymore. Subversion's concept of

"commit"/"update" is a lot more realistic when you have developers which might spend time disconnected from the network. TFS expects developers to always be connected to the server. That's a big minus. I personally find TFS to be less than transparent about how files are organized on the server and on your local disk because of the tight Visual Studio integration. Even TFS's bigger proponents concede that its connected check-in/check-out model is not a compelling option for developers who work disconnected. In a climate where people are starting to look at DVCS options like git and Mercurial over SVN, TFS's "check out" model seems a bit like a dinosaur.

2) **Cost.** Those who say that TFS isn't expensive are either probably very small shops, or are not in compliance with TFS's licensing terms. You need a Client Access License for darn near everything you do. Are you a manager who just manages the bugs? You need a ~$250 CAL (There are 5 included with a retail TFS License). A business user who just wants to report on their issues? A $250 CAL. A developer? $250 (Unless they have MSDN in which case it is included). The server? $500 (Included if you have MSDN). Of course, someone selling you a copy of TFS will tell you that work item tracking is free for additional users, but these additional users can only see the work items which they themselves create, and not the whole team's work items, which isn't too useful in an team-oriented, agile environment. All of this adds up when you have a mid-sized organization, and becomes tough to justify when so many best-of-breed products like SVN and CruiseControl.net's incremental cost is $0. (In

fairness to TFS, though, I'm still waiting for a *really* good OSS issue tracker)

3) **Project structure.** In large teams with a smaller number of projects, TFS will probably work well. If you're a number of small, unconnected or loosely connected line-of-business apps in-house, TFS's structure can start to become overbearing. For one thing, it's not possible to define a taxonomy of projects themselves -- you can set up "Areas" within a project, but all issues and documents are tracked together within the basic context of a "project". Creating new "projects" is often time consuming, and is overkill for small efforts. Of course, SVN has nothing of the sort since it focuses only on source code control, but if you need good small-project flexibility, SVN and another issue tracking tool might be a better choice.

My opinion, for what it's worth:

- For large teams with big, well-budgeted projects, in a Microsoft shop where developers work almost exclusively within the IDE, TFS is the winner. TFS also wins when you need to centrally enforce policy with your projects.
- For a number of small teams, with many varied, smaller projects, or shops where cost is an issue, or teams who have developers who work disconnected from source control, go with SVN.

Share Improve this answer   edited Sep 15, 2011 at 17:44

Follow

answered Oct 25, 2008 at 14:33

Dave Markle
97.6k ● 20 ● 151 ● 171

2 This is exactly the information I was looking for, thanks. – EnocNRoll - AnandaGopal Pardue Feb 17, 2009 at 16:24

..."TFS also wins when you need to centrally enforce policy with your projects." +1. This is huge when dealing with a varying skill set of devs in the shop – StingyJack May 3, 2010 at 19:16

1 +1 for *"still waiting for a really good OSS issue tracker"* - me too... – BlueRaja - Danny Pflughoeft Jun 23, 2010 at 18:30

9 With the launch of Visual Studio 2010 MSFT has included both Client and Server licences with MSDN. So if you have MSDN for all your developers they go FREE and your TFS server is FREE. In addition business users no longer need a licence to create work items nor view the work items that they created. If you have up to 5 business users that want to access TFS then you can buy a retail TFS Server licence for $500 which allows 5 users without a CAL. But additional non-MSDN users will cost you $300 each for a CAL. – MrHinsh - Martin Hinshelwood Aug 21, 2010 at 14:55

3) Project structure: In TFS 2010 you can forgo al of that nastyness for small teams. Just install using the Basic instalation and you get Work Items an Version Control with no Sharepoint... – MrHinsh - Martin Hinshelwood Aug 21, 2010 at 15:09

I'm surprised that someone who has used Subversion in the past would even have a want/need for TFS source

48

control.

My experience with TFS (2005) has been pretty horrible. I've read all kinds of whitepapers & guidance as to how to properly structure your source for various development needs.

Our simple situation, where we have a trunk with mainline development, and integration branch where we integrate changes & deploy from, and a releases branch to keep track of past releases is very common and straightforward, but we are continually running into problems.

My main issues with TFS:

- Merging is a PAIN in comparison to subversion.
- There are unfixed bugs. I ran into one about renaming/merging that has been known for 2 years and a fix will never be released for 2005. We ended up moving our branch to a "broken" folder and we ignore it now.
- Putting read-only locks on your files is friction. Who says I need to edit batch files and build scripts inside of TFS so that it will "check it out" for me? Subversion *knows* which files changed. There are no readonly locks there.
- Speed. TFS is dog-slow over a WAN, and it's really only usable if I VPN into my work computer, which makes my dev experience really slow overall.

- Lack of good command-line and explorer integration. IDE integration is really nice for the day-to-day Get-Latest, adding files, and checking in, but when you need to do things across many projects, it's nice to have good tools at your disposal. And before someone jumps down my throat claiming tf.exe works well... it's not really a cmd line tool. For example, checking in code shouldn't pop up a modal dialog.

...the list goes on. I think even with all of the integration, there are free alternatives that are far superior.

Share Improve this answer Follow

edited Feb 3, 2010 at 8:17
Dave Markle
97.6k ● 20 ● 151 ● 171

answered Aug 29, 2008 at 20:19
Ben Scheirman
40.9k ● 21 ● 103 ● 139

---

5 We got TFS for doing branching and merging and it has been a nightmare. Currently we're looking at SVN. – Brian MacKay Sep 18, 2008 at 16:34

'Merging is a PAIN in comparison to subversion." - why do you think so? I switched from TFS to SVN with AnkhSVN and TortoiseSVN and it's been all unpleasant surprises. Can you point out the conceptual differences between the two modules? – Slavo Aug 7, 2009 at 10:11

---

1 SVN seems much smarter about what can be auto-merged safely, and this is usually the type of merging that I have to deal with. For example, 2 users adding files to the same project file. If 2 people add methods to a file in different

areas, auto-merge also works well. If you modify the same *location* then the merge tool has to kick in. Depending on what merge tool you use, this can be hard or medium (or easy if you have Beyond Compare). Take this line, that line, save file, resolve conflicts, done. – Ben Scheirman Aug 10, 2009 at 21:14

I disagree about SVN merges. There are known issues if files have been edited in both a branch and trunk when you try and do a branch to trunk merge. We spent a weekend at the office last year doing a very complicated merge by hand. That being said I have no idea whether TFS is better or worse. I also wonder why the gold standard of professional source control, Perforce, has not come up.
– Steve Severance Nov 5, 2009 at 1:40

5 Umm, no sorry. I consider myself quite competent in other source control systems, so making a comment about lack of SOC is not at all correct. Last time I checked, solution/project files are changed BY EVERYONE JUST ABOUT EVERY CHECK IN. TFS Should be able to handle merges in these files blindfolded. Yet still, (in TFS 2005) I had almost daily issues with this. It IS THE TOOL. Try anything but TFS and you'll know. 2010 might be better, but why wait around? Git has been solid for me since day 1. – Ben Scheirman Aug 23, 2010 at 20:02

46

I joined an Open Source project over at CodePlex, recently. They use TFS for their source control and I have to say that it's absolutely magnificent. I'm incredibly impressed with it, so far. I'm a huge fan of the IDE integration and how easy it is to branch and tag your code. Adding a solution to source control is something like two clicks, if you've already got everything configured properly.

Now. Is it worth the hefty price tag? I don't think so. The benefit to working on projects at CodePlex is it lets me get the experience with TFS that I need, in the event that I have to use it somewhere later. If you want good IDE integration for your Source Control, go grab VisualSVN integration package. It's a much, much cheaper investment to get a lot of the same features (free on non-domain computers BTW).

Share Improve this answer Follow

edited Aug 17, 2012 at 16:06
bahrep
30.7k ● 12 ● 108 ● 154

answered Aug 7, 2008 at 0:52
Jeremy Privett
4,455 ● 2 ● 33 ● 35

2 build in tfs, not so good, even if you use it, I recommend running cruise control in tandem so that you can actually be independent of your source control. (i.e. tfs 2005, won't build vs 2008 solutions) – DevelopingChris Sep 8, 2008 at 18:51

1 @ChanChan: TFS 2008's build management isn't bad. 2005's build management was a joke, though. – Dave Markle Feb 18, 2009 at 1:09

11 I recommend using AnkhSVN, it's a nice source control plugin for subversion and works exactly like using tfs...except for workitems and the such, which of course are not part of subversion. – galaktor Aug 28, 2009 at 21:15

1 @Jeremy - What functionality in TFS did you use to tag your repository (from quote: "...How easy it is to branch and tag your code.")? – Russell Oct 15, 2009 at 3:44

13 TFS source control tools are horrible compared to SVN, you can't do almost anything other than the most basic things without having to use the "powertools" (rollback a change is a "power" thing??), which are not integrated in VS. Everybody says that TFS is great and everything, and from what I've seen, this seems to be true for the server, but the client tools are horrible. – Eduardo Scoz Oct 15, 2009 at 13:56

43

We're a VS.NET shop, and we implemented:

1. Bugzilla for issue tracking
2. Apache Subversion as a source code repository back-end
3. VisualSVN Server for managing SVN on the server
4. TortoiseSVN (in Windows Explorer) and AhnkSVN or VisualSVN (in Visual Studio) on the client
5. CruiseControl.NET for automated builds

Cost: $0 Benefits: Priceless

If you're a small team, or not ready to buy into the who TFS process, SVN and open source tools are the way to go.

Share Improve this answer Follow

edited Aug 17, 2012 at 16:07

bahrep
30.7k ● 12 ● 108 ● 154

answered Aug 14, 2009 at 21:11

Andrew Lewis
5,256 ● 1 ● 28 ● 32

Same with NUnit and Sharepoint instead of Bugzilla. – boj Jan 19, 2010 at 19:45

1 same here except bugtracker.net on nr 1. – Johan Wikström Jan 25, 2010 at 10:04

1 Same here with TeamCity as build server. Free for up to 20 projects. – ThiagoPXP Apr 13, 2012 at 5:56

VisualSVN 3.0 is free on non-domain computers (the community license) so AnkhSVN is not the only 'free' option :) – bahrep Aug 17, 2012 at 16:03

23

As others have pointed out, TFS gives you a lot more features then SVN does in the form of project management and such. Having used both, and worked with very large companies in implementing TFS, here's my two cents.

1) If you are using TFS 2005, upgrade to TFS 2008. You'll thank me. There are a ton of improvements in TFS 2008 that make it workable.

2) If you live in Visual Studio and you want the IDE integration, go with TFS. I've used SVN integration and almost always drop back to using TortoiseSVN.

3) If you like the idea of accounts being integrated with Windows Authentication, go with TFS. The manageability from that end is nice. There may be hooks for SVN - I'm

not positive, but if you like the GUI driven management, TFS is hard to beat.

4) If you need to track metrics or have easier ways of implementing things like check-in policies, go with TFS.

5) If you have people who won't implement it if it isn't MSFT, go with TFS.

6) If you do more than just .NET (Java work, Eclipse, etc) go with SVN. Yes there are very good products out there (like Teamprise) that work well with TFS. But unless the other languages are a small part of your shop, just stick with SVN.

Outside of that, the SCM features of both are about equivalent. They both do branching and merging, the both do atomic check-ins, they both support renames and moves. I think for people just getting started with the branching and merging concept, having the branches be visible in Source Control Explorer is nice.

TFS really isn't that expensive ($1200 maybe?). Compared to SVN it is, perhaps. The integration to reporting services and SharePoint is nice, but again, if you aren't using that, then it doesn't matter.

What I'd say is to download the 180-day trial of TFS and give it a go. Run a trial side-by-side. I think you'll be happy no matter which way you go.

Share Improve this answer answered Sep 7, 2008 at 14:00

Follow

Cory Foy
7,212 ● 4 ● 32 ● 34

a lot of this is not quite right - sure, I use TortoiseSVN when I can, but only because it is so awesome its my first-choice tool, even when I have AnkhSVN installed. SVN happily does ADS integration - just use VisualSVN Server. Many, many great bug trackers or PM tools integrate with SVN easily and beat TFS management features. Checkin policies are super easy in SVN, just set a pre-commit hook, it even comes with samples and has many other hooks you can use, that TFS doesn't. But .... point 5 is probably the one that matters most. – gbjbaanb Aug 14, 2013 at 12:33

16

As Ubiguchi points out TFS is not a version control product. Buying TFS with the intention of only using it for Version Control would clearly be a waste of money. TFS is an integrated suite of tools to automate all aspects of Application Lifecycle Management (and pretty much geared to "The Enterprise".

Also per Ben S's post - I don't understand your comment about locks. Locks aren't required in TFS at all. Administrators can configure TFS to work like VSS (features demanded by some "unwise" customers) to "Get-Latest on Checkout" which I believe also does a check-out lock.

But through "normal" use of TFS a "check-out" prompts a user for the lock type - and the default should be "none". A user CAN select a check-out (or a check-in lock) - but it is not required. If you don't want locks, don't use them.

TFS does track which users have check-outs on the server for various both performance reasons (make get-latest faster) and project management (I like to see what developers have files checked out and how long their check-outs are).

I'm not real familiar with SVN (I've never used it) - so I can't comment that "mergeing is worse with TFS" - and haven't hit the merge bug Ben S reported - but I've had great success with branching and merging using TFS.

One use case I know TFS is still pretty weak at is for users who are regularly "offline". TFS is a "Server Product" that assumes the users are connected the majority of the time. The offline experience improved in the 2008 release (it was dismal in 2005) but still has a long way to go. If you have developers who need (or want) to often be disconnected from the network for long periods of time - you are likely better off with SVN.

Another feature to consider for SVN fans who are using TFS is the SVN Bridge a codeplex which allows users to use TortiseSVN to connect to TFS. I good friend and colleague of mine uses it extensively and loves it.

Also the comment about a lack of command line surprises me - the command line tools are extensive (although many require a seperate download of TFS Power Tools

I suspect Ben's comments are based on an eval of the 2005 release which was clearly a "Microsoft V1.0" product. The product is currently in 2.1 with Version 3 coming in the near future.

Share Improve this answer Follow

answered Sep 7, 2008 at 13:31
fuzzbone
2,030 ● 14 ● 14

Not realy a wast of money any more when TFS Client and Server is now FREE with every copy of MSDN!
– MrHinsh - Martin Hinshelwood Aug 21, 2010 at 14:59

But it was true in Sep of '08 when I posted (but good update Martin!) – fuzzbone Apr 1, 2011 at 19:25

10

TFS is heinous. At this point I version control locally using SVN (w/ Live Mesh for backups) because I have some many issues with TFS. The main problem is TFS uses time stamps to record if you have the latest version, and stores these time stamps on the server. You can delete your local copy, get latest from TFS, and it will say all files are up to date. It's a silly system that gives you no guarantee that you have the correct version of files. This results in numerous annoyances:

- TFS needs to be informed when any file is edited, so you need to be connected to the server at all times.
- TFS get confused if you edit files outside of the IDE. Further it sets all files to readonly in NTFS.

While TFS supports merging, it's really a checkin/checkout system. If you edit a file you will often find that it is locked to other developers. There are ways around it, but the system is so convoluted you will always run into the issue. For instance, our developers found that they can get around all files being set to readonly in NTFS by checking out an entire solution, which sets an exclusive lock on all files. I did this a few times because subversion has the same syntax for checkout, which does not acquire a lock.

Finally Team Explorer (the client) is a whopping 400 MB, TFS server requires SharePoint and two days to install. The subversion one click installer is roughly 30 MB and it will install the server for you in under a minute. TFS has many features, but its foundation is so shaky you will never use or care about them. TFS is expensive in terms of the license, and in the time developers will waste ranting on stackoverflow instead of writing code :P

Share Improve this answer Follow

edited Aug 18, 2009 at 15:55

answered Aug 14, 2009 at 21:02

James

1,085 ● 9 ● 16

8

My recommendation, Team System isn't worth the money. I have used both and after using Team System, I tried to find a similar replacement. Basically what you are paying for is the integration and you could argue the customization support, but I have been able to create a Team System replacement with a little bit of time and integrating tools together.

I recently asked a question on what others have done to come up with a Team System alternative. I also list the development tools that I used to create the replacement. Hopefully with this answer and the question that I asked, you can find what works for you.

I am not a Team System hater, I just don't think it's worth the money. It is a very nice tool and if you don't mind paying the price for it, then by all means use it. It was the whole reason I created the replacement I came up with. I wanted the functionality Team System provided.

Share Improve this answer Follow

edited May 23, 2017 at 12:16
Community Bot
1 ● 1

answered Aug 17, 2008 at 2:30
Dale Ragan
18.3k ● 3 ● 55 ● 71

Here is a open source version of VisualSVN called Ankhsvn. Its much better now that collabnet has taken it

8 over.

Share Improve this answer Follow

edited May 27, 2010 at 19:26

answered Aug 29, 2008 at 20:51

Donny V.
23.5k ● 13 ● 68 ● 81

3 I agree! I tried it before and it was buggy compared to now. It really rocks. – EnocNRoll - AnandaGopal Pardue Feb 17, 2009 at 17:00

7

If all you need is source control, TFS is overkill. One of my previous employers had TFS, VSS, and Subversion in their enterprise. We didn't have Active Directory or Exchange Server 2003 in our enterprise, so we ended up creating separate users on the TFS server so developers could use it. We had the same sorts of problems with merging that Ben Schierman mentioned, along with other buggy behavior that pushed us toward Subversion.

Whether TFS is the right call for you will depend in part on your budget, the size of your development team, and the amount of time and personnel available for configuration/maintenance of your solution. If you want the additional issue tracking, work item, and project statistics capabilities that TFS provides, it may be worth your while to look at other alternatives. Products like JIRA (from Atlassian Systems) or Trac integrate well with

Subversion and provide the sort of oversight a project or program manager might at a lower price.

In an ideal environment, with Active Directory, Exchange Server 2003 or higher, and dedicated staff for the repository, TFS is more likely to be a good choice.

Share Improve this answer Follow

edited Jan 9, 2011 at 20:52

answered Sep 8, 2008 at 18:35

Scott Lawrence
7,243 ● 13 ● 47 ● 64

6

I have used both at work and at home. They are both very cool in their own right. The only time i would recommend using TFS though is if you will be using more of the features than just the source control. If all you need is source control you cant go wrong with SVN and this is why.

1. VisualSVN Server That is a full SVN server with a nice plugin to manage it with. It lets you use windows authentication right through the UI. Easy.
2. Tortoise Its tortoise, enough said.
3. ankhsvn It is a great SCC plugin. For those that want full VS IDE integration the latest version is a full SCC plugin. So you now get full integration for free.

The above set up is 100% free and will get you through anything you need for source control.

Share Improve this answer Follow

answered Sep 8, 2008 at 18:46

user5119

4

TFS isn't just about Source Control. If you use the whole package that TFS offers, bug tracking, builds, reports, etc then TFS is a pretty solid choice (certainly better than Rational). TFS also integrates well with Active Directory.

Though if you are just talking about SCM, then I prefer SubVersion. I don't really like IDE integration. I also like SVN's convention of Trunk/Tags/Branches structure, and relative ease of switching between branches. Merging seemed easier in TFS though. Tortoise's UI beats TFS's hands down though, especially in regards to adding a file to a repo.

Share Improve this answer Follow

answered Aug 29, 2008 at 21:27

swilliams

48.8k ● 27 ● 102 ● 130

3

I'd say it really depends on your needs. TFS is very nice, I've used it extensively, but it's very much aimed at the enterprise level, if you don't need all of those features it might not be necessary. If you do need those features (especially branching, scalability, work item tracking, etc.) they are worth every penny. Keep in mind that TFS includes bug tracking, work item tracking and other features beyond source control. If you have multiple branches or if you find yourself struggling against some lack of feature or other in Subversion then it might be a good idea to switch. But barring a good reason to switch you should probably avoid the cost and productivity hit of switching source control systems.

Share Improve this answer Follow

answered Aug 7, 2008 at 5:28

Wedge
19.8k ● 7 ● 50 ● 71

3

Having used both extensively, I think Wedge was on the money in noting "TFS includes bug tracking, work item tracking and other features beyond source control".

However, I can honestly say that SVN and TFS seem pretty equal in regards to scalability, and if anything SVN's source control has the edge on TFS due to its inherent simplicity.

If you want work-item and bug tracking alongside your source control then you either go for TFS or you go with SVN and some other, possibly free, tools such as bugzilla. While TFS does integrate both source control and work-item tracking together I honestly think MS should have given it away free as an apology for abusing so many developers with VSS over the years.

answered Aug 29, 2008 at 10:08

Share Improve this answer Follow

Ubiguchi
3,086 ● 1 ● 25 ● 24

2 Bugzilla is awful! Don't recommend that! – Orion Edwards Feb 2, 2009 at 20:41

Yeah, recommend FogBugz instead, it's quite good. – Jeremy Friesner May 3, 2010 at 19:14

3

I have used both SVN and TFS. Main advantage of using TFS is its tight integration with Visual Studio. Bug Tracking, Task Tracking will all go in one place. And the reports generated for these items will help the stake holders keep informed of the project status.

Share Improve this answer Follow

answered Sep 16, 2008 at 16:06

renil
409 ● 6 ● 14

3

I am working on a project with 5 people and we recently switched from SVN to TFS. The entire process has been a nightmare. We have auto generated code from XMLSpy, and TFS does not recognize files modified outside of VS2008. The TFS Power Tools can scan your checkout and fix this problem but it is a pain to have to remember to use these tools. Another problem we constantly run into is the default merging tool in TFS. It is by far the worst merging tool I have ever used. One would think that TFS would be able to handle basic solution merges but so far that has not been the case.

The built in user interface is very useful, but it also has flaws. If I checkout from my solution explorer, sometimes files are that have been added are not checked out. If I do it from the Team Source Control window it works perfectly. Why is that? I look forward to TFS in VS2010 as I have heard great things about it, and SVN is far from perfect, but I would have expected some of these features to function a little more intuitively.

Adam

Share Improve this answer Follow

answered Nov 5, 2009 at 1:52

user203098

Mnay of you issues are fixed in TFS 2010, but TFS is still a "check-out" system, so if you don't tell TFS that you are at least going to modify a file it will not check it against the server. You would love this if you have millions of files in source control, but I feel you pain for small projects.
– MrHinsh - Martin Hinshelwood Aug 21, 2010 at 15:03

3

TFS is great for project management and tracking, however I feel the source control is not as good as SVN. Here are my beefs with TFS:

Check-in/Check-out model

This is a huge con for TFS source control. Unfortunately, VS automatically checks out items for you, even if you don't want to. I've been in a situation where someone checked out some files and then went on vacation. I was in charge of restructuring the directory structure, but was unable to because a bunch of files were checked out to that person. There is no way in the GUI to undo the checkout, which meant it had to be done one by one in the command line. Or I had to figure out how to write a power shell script for this.

VS is required to do everything

Sometimes I want to edit a text file and check it in. This requires me to startup VS 2010, which is a huge beast, just to edit a file and check it in. Something that took a few seconds with SVN now takes me a minute.

As some others pointed out, files are marked read only if they are not checked out. If you make it writable and edit it outside of VS, TFS won't recognize this. This makes editing something outside of VS annoying. This means, firing up VS, checking out the file, editing it another editor, checking in in VS.

Some operations that were easy in SVN are now a pain

- Maybe I haven't figured it out yet, but I found that rolling back a changeset was very tedious with TFS.
- Adding files to source control, which are not part of a solution, is a huge pain. The TFS source control explorer only shows which files are in source control, not which ones are not (maybe there's a setting somewhere for this, I don't know). With Tortoise SVN,

I could simply press Commit on a folder and select which files to add.

Share Improve this answer

Follow

answered Dec 21, 2010 at 12:36

Mas

4,606 ● 5 ● 41 ● 57

Rollback has to be done on the command line.msdn.microsoft.com/en-us/library/dd380776.aspx
– LeWoody Feb 7, 2011 at 18:39

2

I am currently leading the effort to evaluate TFS at my company against the Rational Suite which is what we currently use. So far TFS 2008 is pwning clearcase + clearquest. The dev environment integration is where it really shines.

Share Improve this answer

Follow

answered Sep 22, 2008 at 2:48

rayray2030

565 ● 1 ● 5 ● 11

7 Thats because ClearCase might be the worst source control system most people will ever use. – Steve Severance Nov 5, 2009 at 1:43

1 Not only that the licensing for Clear Case and the other rational tooling is just ridiculous!
– MrHinsh - Martin Hinshelwood Aug 21, 2010 at 15:03

My 10 cents:

2

TFS2005 was a joke - hard to install and even harder to maintain TFS2008 was stable - easier to installer, simpler maintenance, and automated builds that work. TFS2010 is EPIC! - installation is dog eassssyyy. Management is very easy; it's all a nicely laid out UI. Integrating it with VS2008 isn't so easy since you can't create projects in vs2008 you have to use vs2010 (which is stupid). TFS2010 also allows you to change the sharepoint project location instead of having those awful subfolders of TFS2008. TFS2010 also has tools like a burndown chart that is really useful for project management. It's like TFS2010 is for the whole production team including the clients! It still costs way too much though :(

Share Improve this answer Follow

edited Jun 23, 2010 at 18:23
Dave Markle
97.6k ● 20 ● 151 ● 171

answered May 3, 2010 at 19:10
user220583

2

Also take in mind that TFS requires a LOT of more horsepowers from the server hardware. And at minimum one windows server licenses ofcourse.

Best practice, as our company followed, is to use 2 servers: front-end (with integrated sharepoint), and a dedicated sql server in the back-end (we use an enterprise cluster). TFS can be installed on 1 machine, but should not.

In comparation, our svn server is installed on a virtual linux server with 256mb ram and 1 cpu, and is still several magnitudes faster when doing common tasks like checkout-all. The virtual hardware was the lowest vshpere could assign! Disk is fast though (SAN).

I whould suggest that TFS requires dedicated hardware for atleast $5000, while svn server (on linux) can run with any hardware which is obsolete for current windows based os.

Share Improve this answer Follow

answered Apr 20, 2011 at 18:01

Hardon Hardware

21 ● 1

1

In my opinion it depends on the situation and environment in which the project is done. If you have just a simple, small project, then SVN is great. As already some wrote, VisualSVN integrates nicely into Visual Studio s.t. you don't have to do the checkin/checkout over the native file system.

TFS is great for version control, but even better if you really use all of it's capabilities. In my eyes it really becomes worth if you use - for instance - the work items as your integrated repository for handling customer bug reports, new feature requests and for tracking the progress of your project by managing tasks and the according estimated time, used time and remaining time indications.

What is also really interesting is to use the feature of associating work items with source code checkins. See here for more infos about that.

Share Improve this answer Follow

answered Aug 14, 2009 at 21:17

Juri
32.9k ● 20 ● 106 ● 137

1

We are a small team in the process of migrating from SVN to TFS2010. Our biggest reason to do so is the integration in Visual Studio and the WebAccess for bugtracking, that is now part of the TFS.

@Adam: Hopefully we will have a better experience. Can not tell yet...

Share Improve this answer Follow

answered Mar 23, 2010 at 14:00

Marcel
15.7k ● 21 ● 102 ● 161

1

I've used SVN in the past 3 years (coming from VSS earlier) and recently had to switch to TFS2010. The overall feeling is that it is buggier than SVN and except for the nice integration with the tasks/bugs I don't see it as having an edge against SVN. The speed seems to also be somewhat slower than with SVN.

If I were to choose a sourcecontrol now I would still go with SVN.

Regarding tools: - AnkhSVN Visual Studio Plugin is as good as the TFS source control - Tortoise is a lot better than the TFS counterpart

Share Improve this answer Follow

edited Apr 4, 2012 at 17:50

answered Apr 4, 2012 at 13:49

Bogdan
1,393 ● 15 ● 16

0

TFS is great, if you don't need non-developers, to get to pm stuff.

Our helpdesk needs to be involved in the process, and it just wasn't cutting it.

Also the build management in tfs 2005 at least, is attrotious, and it can't even build vs 2008 slns. I really don't like that my source control choice, affects my deployment choices, this is why my team is not an svn shop.

Share Improve this answer Follow

answered Sep 8, 2008 at 18:49

DevelopingChris
40.7k ● 30 ● 89 ● 119

If it ***just*** based on source control, I'd go with SVN. The AnkhSVN free add-in for Visual Studio has been greatly improved in its new release. Also, you get the source

0

code for SVN and the documentation is great! They changed some arcane things in TFS 2010 Source Control, and without the source code it can be very daunting to troubleshoot. Plus, you are dependent on the MSDN team for pumping out the doc's and they do it on their own schedule and at their own depth.

That being said, *TFS obviously offers much more than source control*. It is an ALM tool. Combining it with work items, reporting, automated builds, gated check-in, automated testing, etc. can provide some very rich value that you can only get with connecting disparate tools with SVN. And of course, having the source for SVN is not a failsafe. I've gotten into scenarios with SVN where it would have still taken weeks to totally figure out what was going on.

So, I recommend you look at it from an ALM perspective and see if your company is going to use all of the TFS features or is going to with a best-of-breed strategy (e.g. JIRA).

Share Improve this answer Follow

answered Feb 7, 2011 at 18:36

LeWoody
3,649 ● 4 ● 27 ● 31

-3

TFS by a mile.

I inadvertently cause too many problems for myself with SVNs file-based approach. Source control problems ive experienced: TFS – 0 problems over 2 years SVN – lost count...

Yes I know the price of TFS factors it out for most companies which is such a shame. MS might have a lot more marketshare (and profit) if they had a reasonable pricing model.

Share Improve this answer Follow

answered Dec 8, 2010 at 3:42

andrew
1,260 ● 9 ● 14

2 What kind of problems? How is this related to the file-based approach? – Sean McMillan Jul 7, 2011 at 14:08

1 i using svn for 7 years. and absolutely no problem detected. – pylover Apr 10, 2012 at 3:56