ALM Mag - Vol. 1, Issue 4 - 2013

ALM Mag.com

Empowering Rapid Delivery of Quality Software

Lean Software Development in a Nutshell

ALM State of the Industry Reports: - which one is right?

Countering 6 typical arguments against an Agile/Lean approach

EclipseCon Review

TFS Admin Reports delivered to your inbox daily

Release Management with TFS 2012

What's New in TFS 2012? – Team Explorer

Visual Studio 2013 Preview It's here

Microsoft ALM Certification – Part 2 of 3

* Lab Manager DEMO

Table Of Contents

Newsflash !!!

12

Features

04 ***Welcome to ALM Mag***
Editor's Welcome

05 ***Lean Software Development in a Nutshell***
Examines the realities of software development and concisely explains the Lean approach to addressing those realities
by Al Shalloway

14 ***Lab Manager Demo***
A speedy demo of Lab Manager with SCVMM and the basics of the Lab Manager Build Template
by Steven Borg

Table Of Contents

15 ***ALM State of the Industry Reports: Which One is right?***

A look at the latest Gartner and Forrester reports focusing on ALM toolsets

by Clementino de Mendonca

19 ***EclipseCon Review***

A review of several of the presentations at the March EclipseCon event

By Keith Pleas

25 ***Microsoft ALM Certification – Part 2 of 3***

Part 2 on Microsoft's new ALM certification focusing on **Exam 70-497: Software Testing with Visual Studio 2012**

by Anthony Borton

29 ***Release Management with TFS 2012***

There are many features in TFS 2012 that support creating a release management strategy

by Martin Hinshelwood

35 ***TFS Admin Reports delivered to your inbox daily***

TFS Admin Reports emailed daily notify and alert on cube status

by Anna Russo

38 ***What's New in TFS 2012? – Team Explorer***

New to TFS 2012… this article focuses on new features in Team Explorer

by Mohamed Radwan

45 ***Countering 6 typical arguments against the Agile/Lean approach***

The arguments against and mitigating rationale for an Agile/Lean approach

by Tomáš Tureček

Welcome

Keith Denham
Publisher

Treasure Cove Publishing
Volume 1 – Issue 4
May/June 2013

This issue coincides with the 5th named version of Team Foundation Server, TFS 2013. This version brings exciting new features and functionality to play that many have been waiting for, particularly those wanting an easy tie from the executive planning level into the development tier.

Al Shalloway examines the realities of software development and concisely explains the Lean approach to addressing those realities.

Steven Borg presents another great Northwest Cadence video demo of Lab Manager with SCVMM and the basics of the Lab Manager Build Template.

Clementino de Mendonça examines and sheds light on the survey findings from the latest Gartner, Forrester looks at the ALM tools available today.

Anthony Borton's article is the second of a three part series outlining the new Microsoft MCSD ALM certification. Part 2 covers:
Exam 70-497: Software Testing with Visual Studio 2012.
This is Microsoft's first exam focusing primarily on Microsoft Test Manager (MTM) 2012.

Martin Hinshelwood tells that to the contrary, TFS 2012 has many features that support creating a release management strategy.

Anna Russo shows hoe TFS Admin Reports can keep us advised and alerted to the state of the TFS cube.

Mohamed Radwan presents the new and interesting TFS features available in Team Explorer.

Tomáš Tureček while countering 6 typical arguments against an Agile/Lean approach actually brings enlightening information to both sides of the table.

Not yet a member? Take advantage of our early bird Offer… click this link for more details. http://www.member.almmag.com/goto/foundingmember

Enjoy!... and Warmest Regards,
Keith Denham

Keith Denham

P.S. Don't forget, this is an interactive magazine, tap on the audios and videos and tap on the highlighted links to see additional related content.
Tap on the Index Page Button at the foot of any page to return to the Index.

Lean Software Development in a Nutshell

Al Shalloway
Founder & CEO
Net Objectives

Much of the work done in software development groups (IT or product) is the result of delays and interruptions. This sentence sounds funny, doesn't it? But on reflection, let's see where a lot of our time (work) is spent:

- Reworking requirements
- Building software to satisfy missed or misunderstood requirements
- Finding bugs
- Thrashing while trying to integrate two components that were built semi-independently of each other

What causes most of this work? I'd suggest delays. In the above cases, respectively:

- Delays in the time from getting requirements until using them
- Delays caused by having to do big batches of work until getting feedback on it
- The delay from writing the bug until it is discovered (fixing is fast if this delay is short, finding gets exponentially longer the longer this delay is)
- The delay from testing the synchronization of the two components

Not to mention inefficiencies create by decisions made at the wrong time (too early means a delay from when the decision was made until it should have been made, too late means a delay from when it should have been made to when it was made).

One hears a lot about batch size in lean because batches of work are the easiest things to spot. But it is actually the delays and interruptions we are trying to eliminate when we manage flow. Managing batches does this because the underlying cause of batches (too much work in progress – WIP) is the same underlying cause of delays and interruptions.

Lean's mantra of "delivering fast" is often misunderstood to mean "go fast." It doesn't mean that. It means "avoid delays and interruptions in order to get there (delivery) fast." You avoid delays and interruptions by attending to flow.

How to Read this Article

It is best to read this article with a fresh mind. If I had a Men-In-Black neurolizer I would use it on you to wipe out what you've heard about both Lean and Agile. There is a lot of nonsense out there about what each of these mean. On the Lean side we've heard many say:

- Lean is about being more efficient
- Lean is about needing fewer people so you can let some go
- Lean is about eliminating waste
- Lean is based on manufacturing methods and is best learned from Toyota

On the Agile side we've heard many say:

- Agile requires you to release every 2-4 weeks
- Agile says to not do up-front design
- Agile says to not look ahead for planning
- Agile says not to document anything
- Agile says teams get to do what they want
- Agile only works for software
- Agile does not have a place for management

<flash> <blink> Please forget you ever heard any of these. J

Where We Are

In order to make a correction in direction it is important to first understand where one is and sometimes how one got there (so we won't keep coming back here). Most software organizations are currently overloaded with work. They have too many, too large, projects that are poorly prioritized being worked on at the same time. Most of these projects are bigger than they should be for several reasons. A prevailing one is that when the stakeholder's project comes up to be done, everything that is anticipated as being needed is asked for because it is unclear when there will be a second chance (and the change process to do this is very painful). So everything is asked for - we call this "the getting it while the getting is good syndrome."

The result is that people work on multiple things. It is not unusual for key people to be working on six or seven projects. When we investigate organizations that claim their folks work on one project, we discover that, while they are working on only one project, they are likely working on six or seven things within that project. The resulting delays/interruptions in their workflow are very similar to folks working on multiple projects.

Working on multiple projects also causes many interruptions. When information is needed, the person who has it is often not immediately available – creating a delay for the person who needed it in addition to the interruption of the person who has the information. Critical people are typically interrupted the most, meaning your best people are reduced the most in efficiency. There is another side effect of all of this: the development group gets so jammed executives can't wait for their pet projects to be done so they interrupt entire projects by interjecting new ones.

It's like in the picture to the right. We know we're overloaded but we'll still jam in one or two more things into the spots available. In the hope we'll get more done we slow everything else down. The root cause of our problem is our focus on efficiency and utilization. Lean talks about eliminating waste, but sometimes we can't see the waste. In software, writing and fixing bugs looks the same as writing software without bugs. It's hard to tell what's happening when it is going on. Software is different than manufacturing; therefore, we need something different to look at to improve our methods.

People often think the downside of all of these projects is task-switching. We know task-switching is bad, but what can we do? The reality is that task switching is only a (small) part of the problem. The bigger problem is that delays and interruptions themselves create the additional work I mentioned at the start of this article.

We are in an endless cycle of having too many things to work on which causes delays, and interruptions which vastly expands time to market which tends to have people interrupt us and to ask for more than they absolutely need because they are afraid there won't be a another chance. This cycle just makes things worse and worse. It cannot be fixed with the current orthodoxy of achieving high utilization, but instead is a downward spiral. Adding work to an already overloaded team just makes it worse.

The Lean Mindset

Lean is based on two fundamental principles – first, one must take a systemic view to improvement and second, one must do this by looking at the flow of work though the system. The first is a shift from managing people to improving the systems within which people work. The second is a shift from focusing on productivity directly to removing delays in workflow to increase productivity.

Systems Thinking: The foundation of Lean

Systems thinking means that you recognize that a system is not merely a collection of its pieces but rather includes the way they work together. That is, one must take a holistic view when trying to improve the system since affecting one part of the system will have an impact on the other. Lean takes this a step further by incorporating Deming's belief that the system within which people work has a very profound impact on the quality of the work these people do.

This does not mean to imply that people don't matter. It does mean, however, that the system within which people work is as significant as the people. One way to look at it is the belief that putting good people into a bad system will result in poor results. It isn't systems that make people great, however. Putting poor people into good systems won't get good results either. But the attitude is that most people are competent and motivated and that if you are not getting results you want you need to improve the system in which the people work instead of focusing on how to get the people to be more motivated.

Lean focuses on getting good people to create great systems to create spectacular results.

A focus on time

Lean's focus on time is the main insight that extended Deming's system thinking and turned our focus away from utilization. Lean suggested a focus on doing things at the right time to improve time eliminate waste and get products out the door quickly. Instead of looking at the cost of each step we look at the cost of delaying the product. Think of the development of a product from its start to its finish. We call this "concept to consumption." There are times we work on developing the product and times people are interrupted in this workflow and work on something else. The more delays and interruptions in this workflow the long it takes to get the product out the door. This is another way of looking at "time to market" an old concept, of course. Lean software's twist on this was that these delays and interruptions create additional work for us to do. Therefore, by attending to removing delays and interruptions, we can deliver faster because we have both fewer delays and less work to do. We mentioned a few of these delays at the start of this article.

In other words, a project that could be built by a focused team in one-month will take more than two months if they work on it half-time. Working on multiple projects not only extends the time required but adds work to be done. The extra time is not due to task-switching, but due to new work being literally created from the delays of working on multiple projects.

Lean therefore suggests that removing delays and interruptions in the workflow increases productivity, lowers cost and improves quality. Lean suggests that it is these delays and interruptions that are the source of waste and poor quality. Therefore, if we can eliminate these delays, we can improve our quality, lower our cost, raise our productivity and delivery product quicker - not by going faster, but by removing interruptions and delays. This is a different focus than most companies have.

The Source of Working on Too Many Things

The main cause of working on too many things are:

- Long (annual) planning cycles
- Larger than necessary projects
- Doing development in large batches predicated by phase gate methods
- Specialized team members who have to help out on many projects
- Validation of what has been built in large testing cycles at the end which results in much being worked on at one time – which results in late feedback

Implementing Lean Software Development

Lean Software Development tells us we need to:

1. Limit work to capacity to remove delays and lower interruptions
2. Attend to the timing of the workflow with methods such as acceptance test-driven development, test-driven development and continuous integration
3. Use time-to-market (which Lean calls "cycle time") as the ultimate measure of progress

The solution is to focus on the most important functionality, get it completed and then go on to the next most valuable work. The question remains, of course, how to do this. Lean Software Development tells us to look at our workflow and attend to the cost of delays. The image of our work flowing through the system gives rise to the name "value stream." That is, it is the flow of value from "concept to consumption."

This is accomplished by working on smaller pieces of work. This, in and of itself, has a very beneficial effect. By necessity, timeframes shorten when smaller pieces are worked on. Eli Goldratt put it this way, "often reducing batch size is all it takes to bring a system back into control".

To have a value stream which can deliver products quickly we must do the following:

- Identify the most valuable work to be done
- Avoid building what's not needed
- Provide work to teams in a manner that allows them to work together with little delay
- Have the workflow be in the proper order
- Evolve robust system and application architectures using emergent design
- Manage the workflow so all of these actions are appropriately load balanced throughout the value stream

A brief look at each one

Identify the most valuable work to be done.
This first requires the use of Minimal Marketable Features (MMFs) (or their equivalent) in identifying the right sized, most valuable, capabilities to be delivered. These can then be prioritized prior to entering the development process.

Avoid building what's not needed.
One often sees functionality built in large blocks. This is often done to create economies of scale – particularly in the test environment. Unfortunately, this also often results in functionality being built that was not fully understood or needed. Furthermore, it is often difficult to take advantage of what has been learned during this process and therefore new ideas, that are often more valuable than what was originally planned, have to be postponed for the next release.

Provide work to teams in a manner that allows them to work together with little delay.
The biggest shift to Lean-Agile requires that teams pull work from their backlogs of work to be done when they are ready. This enables them to achieve stability in their development methods by avoiding them from working beyond their capacity. When more than one team is required to work together, these backlogs must be populated in a coordinated fashion so teams work in a coordinated fashion.

Have the workflow be in the proper order.
Recent advances in Agile engineering practices (TDD, ATDD, BDD, continuous integration) suggest that using tests as a validation method greatly increases quality and

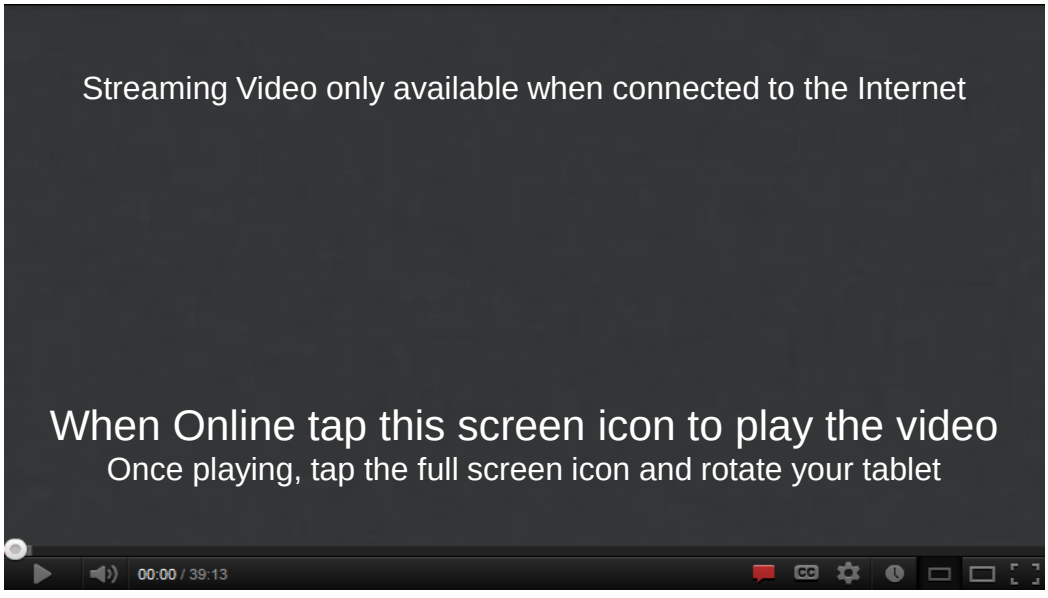
helps avoid misunderstandings of requirements. While not originating from Lean, they are very consistent with the Lean concept of reducing delays – particularly from initial communication of a requirement until its validation. Lean teams must incorporate these ideas to avoid delays and unnecessary rework.

Evolve robust system and application architectures using emergent design. Architectures have always needed to evolve. Attempts at creating an architecture that will not need to change have at best put off the inevitable. It is better for an architecture to be able to change than it is for it to be able to withstand change. Fortunately, the use of design patterns, refactoring, and test-driven development can be used to create these architectures.

Manage the workflow so all of these actions are appropriately load balanced throughout the value stream. It is essential to manage the work in progress (WIP) throughout the work flow. Exceeding the capacity of the teams at different stages of work will result in queues, delays and interruptions.

Lean, Management, and Visibility of Work

While there is often debate about whether you can manage something you can't measure, everyone agrees that you can't manage something you can't see. Lean suggests that we map out our workflow in what are called visual controls – that is, visual representations of our work that enable us to manage the work. As a practical matter, these value stream maps can be created as Kanban boards. See the following short video…



These workflow representations (Kanban boards) can be used to create a context within which all parts of the value stream engage. This avoids local optimizations from hurting true productivity. The ultimate measure used by management in lean implementations is value delivered by time. Management's role is to help create the bigger picture needed for teams to work together and to remove any impediments in the overall workflow.

Push vs. Pull

The essence of effective Lean-Agile development is to deliver the most important capabilities in the timeliest manner. To enable developer groups to respond quickly, they must work without slowing themselves down with delays and interruptions. This means they need to accept work at their own pace. This approach is called 'pull' because they pull work from a backlog of prioritized work. This enables a more stable rate of productivity. This, in turn, creates greater predictability of the rate at which they accomplish their work. On the other hand, when work is given to them with set schedules, the set schedule typically requires more things being worked on than should be. This contributes to delays and interruptions causing additional work. This instability in the development process causes unpredictability of what they can accomplish.

Not the Whole Story

This, of course, is not the whole story of Lean. Lean provides insights into product prioritization, management, and learning.

Summary

Much of the work done in software development is not value added work but is work caused by delays and interruptions. Working from old requirements, re-working requirements, the time to find bugs, thrashing due to integration, implementing misunderstood requirements often account for more than half of the work done by a development team. Delays in, and interruptions to, our workflow is mostly due to working on too many, too large projects. We must start at the beginning of the value stream by doing proper product portfolio management. We must create visibility throughout the value stream so we can improve the efficiency of the work being done. We must manage the workflow across teams when more than one team is involved. We must shift from focusing on isolated optimizations of efficiency to a holistic view of the entire workflow.

Cause of Delay	Cause of Efficiency
• Big Planning Cycles	• Small planning cycles
• Big projects	• MMFs
• Implemented in big Batches	• Incremental development
• Specialized team members	• Cross-functional, load-balanced
• Validation at end	• Quick validation

Table 1. Causes of Delay and Actions to Counter Them

Al Shalloway is the founder and CEO of Net Objectives. With over 40 years of experience, Al is an industry thought leader in Lean, Kanban, product portfolio management, Scrum and agile design. He helps companies transition to Lean and Agile methods enterprise-wide with consulting and training. Al is a SAFe Program Consultant the co-author of four books on Lean, Scrum, Design Patterns and Agile Design.

Visual Studio 2013 Has Arrived

On June 26th Steve Ballmer announced the Visual Studio 2013 Go-Live preview at the Build summit. On top of the already feature rich quarterly releases this named version provides great new features and functionality we can all put to good use.

Agile Portfolio Management

A major focus with this release that provides all levels of the Enterprise visibility into their view of the project with Backlogs that now start at the highest business initiative level and provide both upward and downward "view mobility" through a reported 7 levels of hierarchy. This is part of a continuous effort to build out enterprise agile capabilities.

Version Control

There are too many changes to cover in this space so we'll focus on a few of the top things.

There is a new visually appealing Team Explorer home page that provides easy navigation to your web based task board and the list of solutions in your workspace.

To those who yearned for the pre-VS11 style Pending Changes window, Microsoft listened and introduced "Pop-out Team Explorer pages" that with a simple click allow you to pop-out the page and dock the pending changes window anywhere you want. You can also "pop out" other pages too (like the build pages) with more planned to be added by RTM.

The Git innovations you seen recently on Team Foundation Service have been added and will ship in the on premises TFS in TFS 2013.

Coding

Coding joins Top Gun with a new "heads up display" feature in Visual Studio that provides key insights into your code as you are working. These "indicators" help you to learn more about your code and are but the first of many planned.

Memory diagnostics and memory "snapshots" enable you to find memory leaks in production. Two snapshots can be taken and compared to see what objects changed.

Testing

Visual Studio Testing capabilities continue to evolve and in VS/TFS 2013 are raised to a new level with more new capabilities. Test case management now allows you to more fully manage your test plans. You can now create/modify test plans, suites and shared steps on the web without having to switch to the Test Professional client.

With Cloud load testing you can now load test your apps without configuring any infrastructure. Use Visual Studio Ultimate Edition to create a load test and point it at Team Foundation Service and say Go! The number and size of runs available is limited until the service releases.

Release Management

Microsoft's acquisition of InRelease, a great release management solution that's been natively built to work well with TFS, allows you to manage all of your in-flight releases. For each application and each release you can define paths in an automated deployment pipeline that have stages, acceptance criteria, approvals etc. This functionality will seamlessly integrate with TFS and VS and is planned to evolve even more over time.

Team Collaboration

The always "on" Team Rooms are durable collaboration spaces that "permanently" record everything happening in your team. Notifications – checkins, builds, code reviews, etc are all configurable and reside in the Team Room as a living record of the activity in the project. A Team Room also serves as a conversation space with the rest of your team.

Conclusion

This is but a few of the highlights of the new Visual Studio 2013. There are many more to cover in the issues ahead.

Click on the banner above to download your Visual Studio 2013 Preview.

NOTE! This Preview is covered by a Go-Live license and is fully supported for production use.

Look for more information about VS 2013 in these pages in the coming issues.

The following are some more links to good information about VS 2013 Preview:

http://www.microsoft.com/visualstudio/eng/2013-preview
http://blogs.msdn.com/b/bharry/archive/2013/06/03/visual-studio-2013.aspx
http://blogs.msdn.com/b/somasegar/archive/2013/06/26/visual-studio-2013-preview.aspx

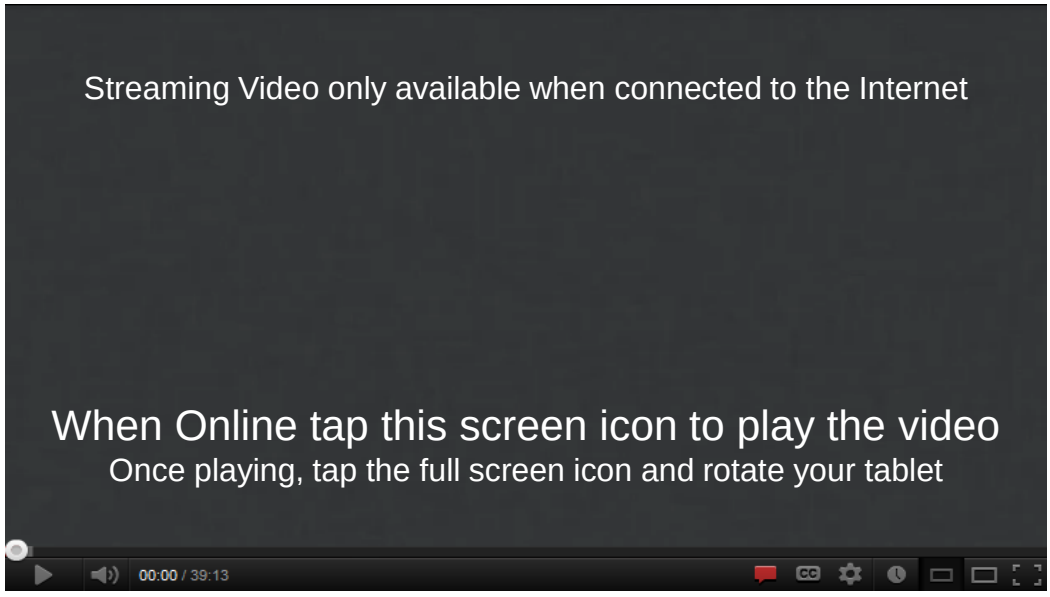
Lab Manager Demo

Steven Borg
Co-Founder
ALM Consultant
Northwest Cadence

Interested in using Lab Manager with SCVMM? Not sure how to test your installation? Check out the demo below! Using Standard Environments is fairly straightforward, but many people have challenges validating that they have a correct Lab Management infrastructure when using a System Center Virtual Machine Manager (SCVMM).

In this Lab Manager demo, Steven Borg will cover several components in Lab Center, how a minimal architecture is defined for a SCVMM implementation, and how to verify that all the pieces are working effectively together. This speedy demo also includes a quick walk through the basics around the Lab Manager Build Template and how to use a Team Build to create a simple environment build test.

Recommended Audiences
Business Decision Makers, Developers, Technical Decision Makers



Steven Borg is Co-Founder and Strategist at Northwest Cadence. A Microsoft ALM MVP since 2005, Steven speaks regularly at software development and Visual Studio conferences, has authored Microsoft courseware and white papers on Team Foundation Server, Scrum and lean software development. Steven brings successful lean and agile adoption to companies currently using traditional project management techniques.

NORTHWEST CADENCE® where technology meets teamwork

Northwest Cadence, Inc. *provides training and consulting services across the United States. The exclusive focus on Software Process and Microsoft ALM tools ensures clients benefit from deep technical expertise, sound process acumen, and hundreds of practical experiences on ALM projects. They partner with software development teams to improve processes and business results.*

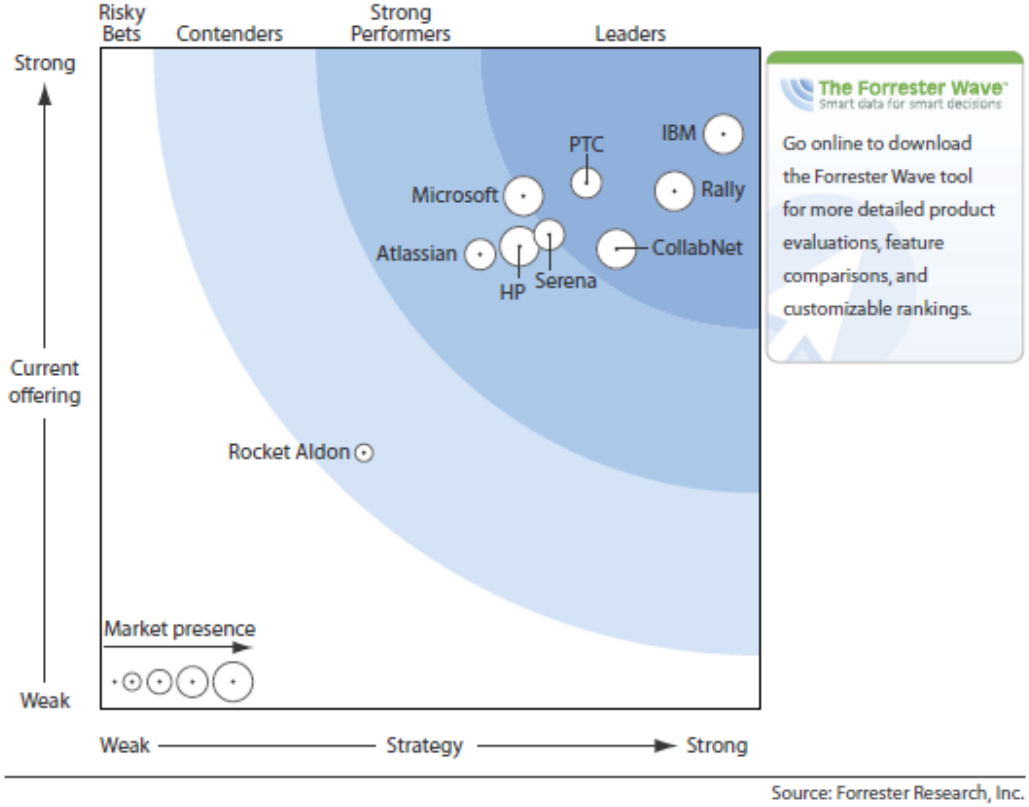
ALM State of the Industry Reports: which one is right?

Clementino de Mendonça
National ALM Architect
Neudesic

Over the last 6 months I have been asked which one is better: TFS, based on the latest Gartner report, or Team Concert and Rally, based on the latest Forrester report (which you get here and here)? The answer is quite simple if you actually read both. Here is the short version:

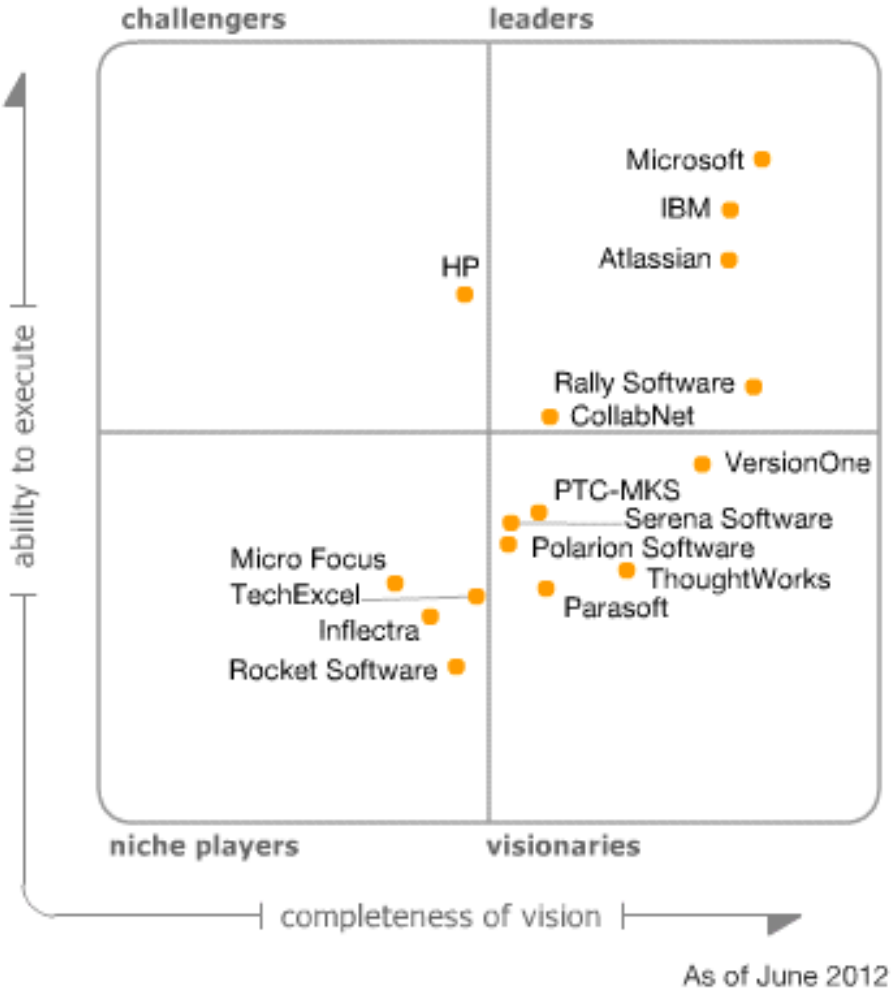
Forrester's Wave report is made every two years, and it was comparing the latest version of Rally and Team Concert with Visual Studio/TFS 2010, so naturally TFS is in the second tier as the Agile backlog and planning tools were not considered;.

Figure 3 Forrester Wave™: Application Life Cycle Management Tools, Q4 2012



Source: Forrester Research, Inc.

- Gartner's Magic Quadrant report compares Rally and Team Concert with Visual Studio/TFS 2012, so the three are in the leadership tier. In fact Microsoft is the leader at this moment, a remarkable achievement considering that it was a late entrant in the race. Team Concert and Rally continue almost neck to neck, and it will be interesting to see what the next report will tell us.



Both reports are useful because they also take into account other ALM tool providers. Also the combination of the two outlooks tells us about the complexity of the ALM industry, which is not captured by any single report.

That said, in the specific case of TFS you will need to pay close attention to the Gartner report as it is based on the latest and greatest. Also Forrester worked with Microsoft to create a new specific ROI report based on TFS 2012. It's the Forrester "Total Economic Impact Of Microsoft Application Lifecycle Management", and it shows amazing ROI over a 3 year period.

I also recommend reading the excellent Ovum Technology Research Report: Software Lifecycle Management 2011/2012. It provides yet another profile of the leadership tier ("Shortlist" in this case) which includes TFS as well, even though the report was based on TFS 2010.

Rating	Company/Solution	Ovum Opinion
Shortlist	HP	HP has recently launched a new ALM solution that offers comprehensive end-to-end functionality, with exceptional strengths in enterprise QA and test management.
	IBM	IBM Rational is steadily rolling out its next-generation products that support its Jazz framework, while building on its existing base with managing the lifecycle of software embedded in complex systems of systems.
	Microsoft	Microsoft's Visual Studio 2010 continues to provide .Net developers (and others) a full featured ALM platform. The solution further enhances capabilities with Test Lab management and tie in to the Azure cloud.
	MKS	MKS pioneered development of unified tooling covering SCCM, requirements, quality, and project management; a key focus is product engineering companies managing development of software in complex systems.
Consider	Micro Focus	Micro Focus has taken the first steps in converging the QA products inherited from the Borland and Compuware testing tools acquisitions, but has yet to roll out broader based offerings that reflect its existing tools portfolio.
	Polarion	Polarion has steadily built out its ALM solution organically, based on open source solutions, to provide an enterprise level ALM solution, with the systems/product engineering market a particular focus.
	Serena	Serena is evolving into a vendor of ALM solutions that are based on a process orchestration engine. It is also partnering with Nolio to develop a unique release management solution that extends into orchestrating software deployment.
	TechExcel	TechExcel's workflow rooted solution and knowledge base system marks it out as an exceptionally well integrated solution, with increased support for SaaS ALM a 2011 target.
Explore	Atlassian	Atlassian has built its development management solutions in a unique way, offering some industry leading products such as JIRA. Its cloud based studio and recent acquisition of bitbucket add further diversity.
	CollabNet	CollabNet has built a growing stable of tools leveraging its popular Subversion SCCM technology. The Danube acquisition has extended its solution covering Agile planning.
	Rally Software	Having helped pioneer cloud-based ALM, Rally has built an Agile planning solution that takes a lightweight approach to related areas of project, requirements, and quality management.
	ThoughtWorks Studios	ThoughtWorks Studios offers its Adaptive ALM solution to ensure teams can perform in an Agile way or adopt hybrid practices. A vendor at the forefront of Agile thinking; for example Go addresses DevOps needs.

Index Page

What I like about this report is its Ovum SLM Solution model, which highlights new trends such as DevOps and the growing overlap of ALM and PLM (Product Lifecycle Management). It also includes a list of vendor profiles such as Seapine and Tasktop, and their current ALM products. This is very useful if you are trying to understand industry trends as opposed to tracking a single company.

Notice how the same company/ALM suite can be in a completely different ranking depending on the report focus. That's why I recommend paying close attention to the methodology used by each report vendor. The best way to understand the complexity of the ALM industry is to create your own composite view based on many sources. At a minimum you will need these four reports.

Just reading the marketing perspective from each company, which obviously will leverage those reports to provide a biased perspective of their own products, will put you in a position of an uninformed bystander in the ALM world.

Clementino de Mendonça is a Professional Scrum Developer Trainer, a Neudesic National ALM Architect, a former National Lead for ALM and Testing with Capgemini/Sogeti, and was on the Microsoft Services National ALM Team for 4 years. On the Visual Studio Process Team in 2007, he helped design the Visual Studio 2010 process templates (MSF Agile 5.0 and MSF CMMI 5.0).

EclipseCon Review

Keith Pleas
Organizer and
Conference Chair for the
ALM Summit

The **ALM Connect** conference in March, 2013 – co-located in Boston with EclipseCon – presented a variety of ALM sessions covering topics from Scrum to testing, and focusing on many of the Open Source tools that make up the typical Eclipse-based portfolio.

Lifecycle Integration

Several sessions also covered integration techniques, including "Lifecycle Integration – The secret sauce of ALM success" by ALM industry veteran Dave West, now with Tasktop. Dave's session described how the common focus on vendor-based tool suites has removed focus from the end-to-end business process, and how the tool silos are becoming more entrenched as developers and teams are empowered to select the tools that make them most productive.

A recent survey of 280 customers by Forrester Research asking "Which would you consider as major roadblocks of ALM solutions?" showed that integration has become the #1 problem in ALM:

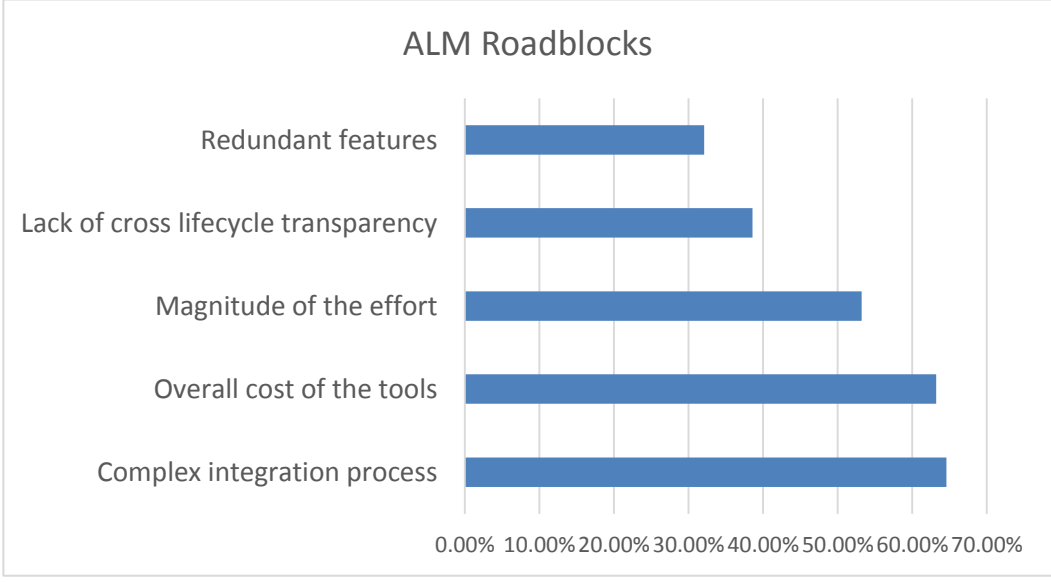


Figure 1: Reported ALM Roadblocks

Tasktop's solution for getting the data out of the tool silos is to practice Software Lifecycle Integration (SLI), which utilizes an Application Lifecyle Bus (ALB) that functions - similar to an Enterprise Service Bus (ESB) - to connect each of the stages of software delivery:

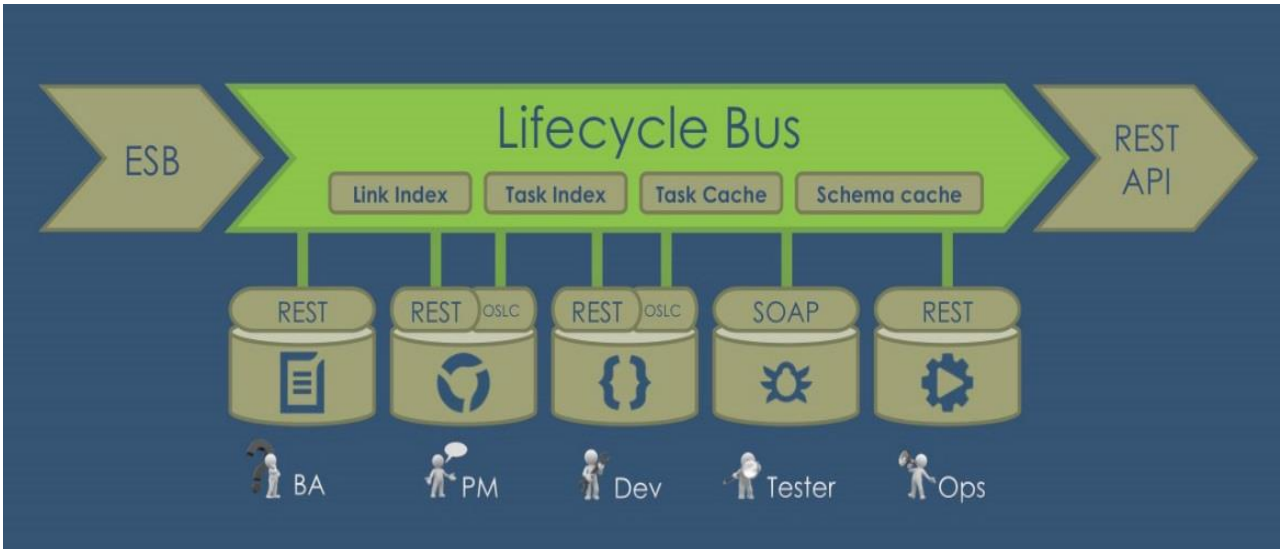


Figure 2: Application Lifecycle Bus

Also, enterprise architects who haven't exactly been "feeling the love" of the agile software movement will be gratified to note that Tasktop has identified an emerging Lifecycle Architect role to define the architectural concepts used to plan SLI deployments.

OSLC

Much of the tooling that connects the different silos utilizes plumbing built on the Open Services for Lifecycle Collaboration (OSLC) specification, which was covered by Steve Speicher and Michael Fiedler – both with IBM Rational – in a session titled "Leveraging W3C Linked Data, OSLC, and Open Source for Loosely Coupled Application Integration".

As explained by Steve and Michael, the custom solutions for communicating between the silos that was common circa 2005 was less than ideal, and did not provide a common system for governance, metrics, and reporting "adored by managers and philosophers and divines" (to mis-quote Emerson). Around 2010, the emergence of REST-based protocols and Linked Data (as defined by Tem Berners-Lee in 2006) provided a sufficient framework for all of the lifecycle systems to communicate together.

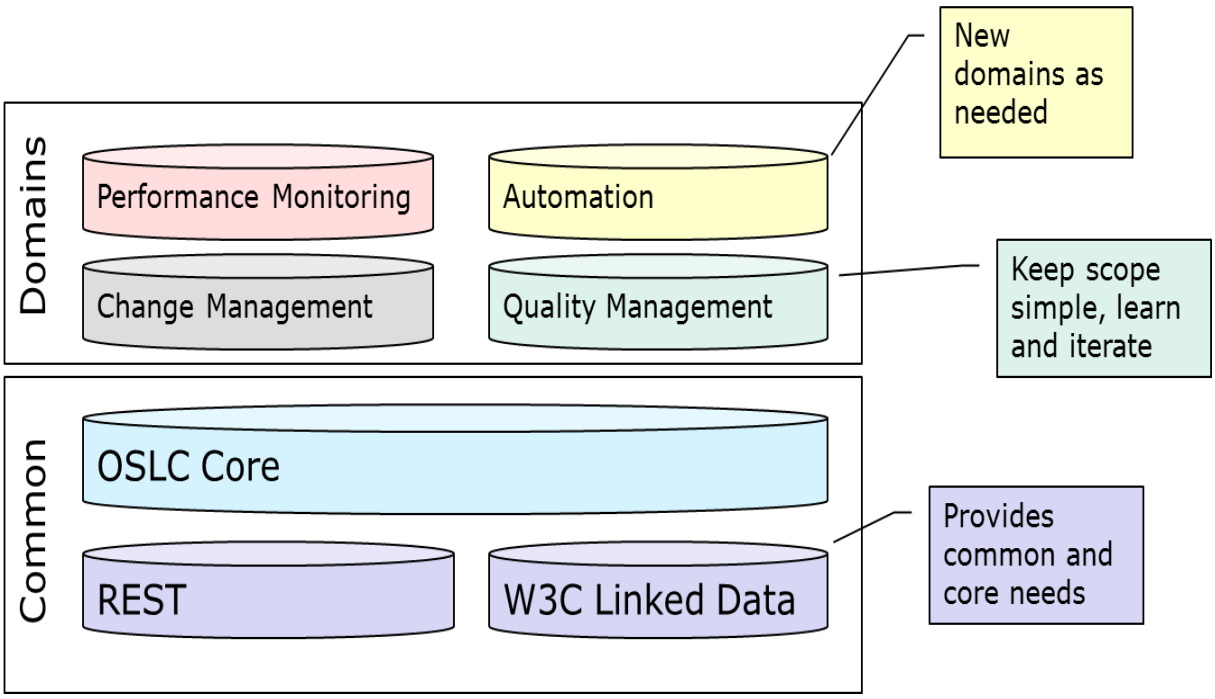


Figure 3: OSLC Layered Model

The OSLC Steering Committee has taken steps to move the OSLC specification to OASIS, the international standards consortium. Working independently of OSLC, the W3C last year launched the Linked Data Platforum (LDP) Working Group and is anticipated to deliver a W3C Candidate Recommendation by the end of 2013.

Executive Day, organized by Dave West

Dave West also organized "Executive Day" at ALM Connect. While the day itself was by private invitation, all of the videos and presentations are available on a dedicated Executive Day area on the Tasktop site (free registration required). Many of the speakers at this elite event have also presented at past ALM Summits, including Israel Gat, Sum Guckenheimer, Ken Schwaber, and Mik Kersten This was the agenda:

- Welcome from Eclipse (Mike Milinkovich, Eclipse Foundation)
- The changing face of ALM with Mobile and SaaS (Jeffrey Hammond, Forrester Research)
- What ALM knowledge you can expect from Computer Science Graduates (Gary Pollice)
- Managing Complex Supply Chains with ALM (Mik Kersten)
- The User is the Center: Apps in the World of Engagement (Raziel Tabib)
- Agile 2.0 Software Development in the Era of the Social Graph (Israel Gat)
- Scrum – Success Ends with Middle Management (Ken Schwaber)
- Future of ALM Panel (Sam Guckenheimer; Jeffrey Hammond; Mik Kersten; Raziel Tabib; Mike O'Rourke)

Enterprise Scrum

Ken Schwaber - co-creator of Scrum and CEO of scrum.org – talked about companies that have been large-scale adopters of scrum. While it can now be taken for granted that every organization wants to the “agile”, the most success has been achieved when management – particularly senior executives – are directly involved in the agile adoption process rather than pushing responsibility down into the software development team itself.

Of course, management will feel most comfortable with a metric that shows them progress towards agility, and Ken described the Agility Index built on transparent metrics for frequency of releases, time to get a small change to a customer, and even customer satisfaction.

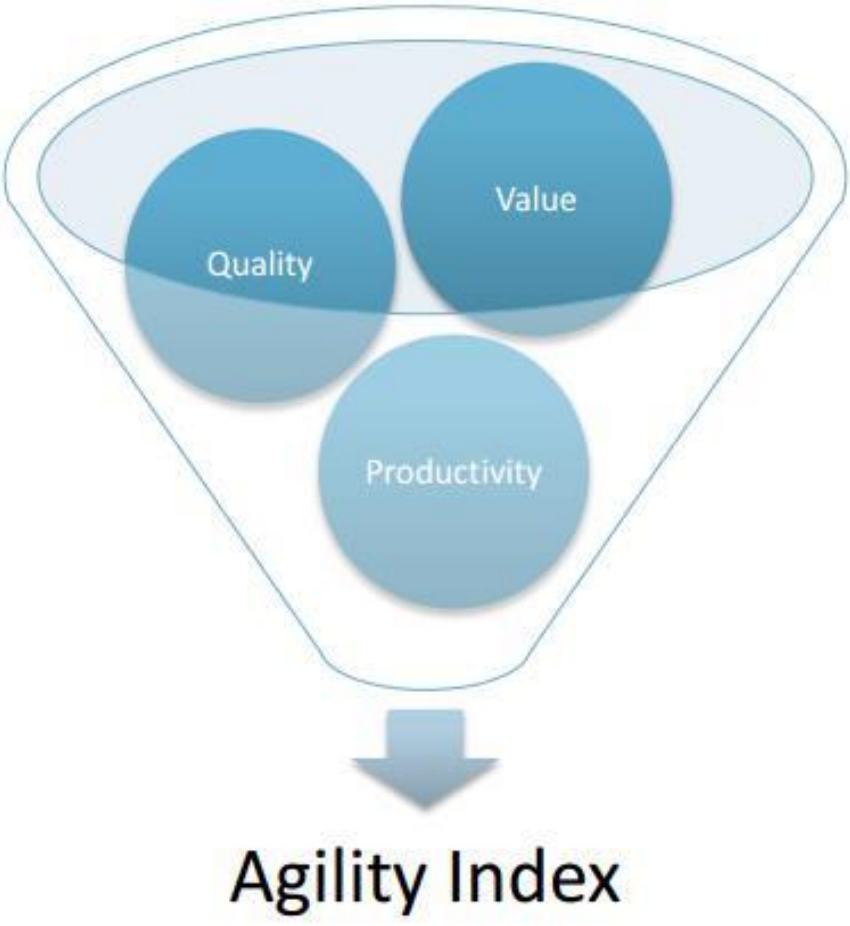


Figure 4: Inputs to Agility Index

Systems of Engagement

Several of the ALM Executive Day speakers referenced Geoffrey Moore's "Systems of Engagement" described in his classic "Crossing the Chasm" books and in the Systems of Engagement and the Future of Enterprise IT: A Sea Change in Enterprise IT AIMM whitepaper. Jeffrey Hammond described how "modern applications" - targeted at multiple devices – have shifted the focus of development from getting software correct and working to getting feedback from users:

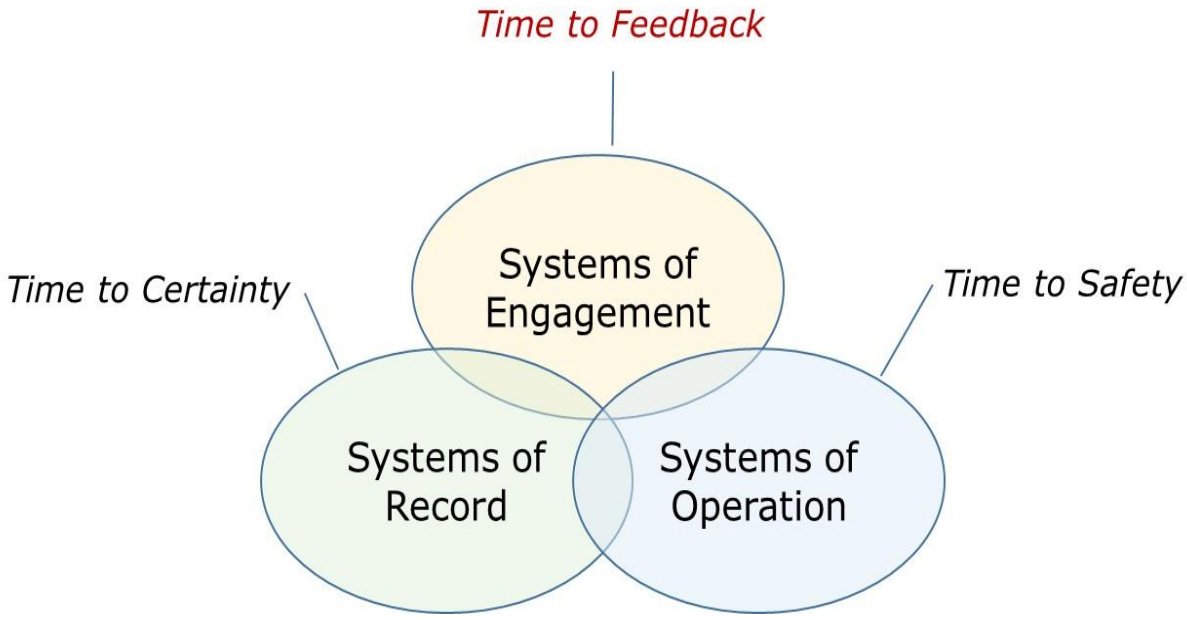


Figure 5: Systems Model Values

Jeffrey describes a process based on agile principles that uses personas, journey maps, wireframes and prototypes, and Kanban boards to drive a process based feedback based on analytics build into the software rather than requirements documents.

In our increasingly complex environment, the implications of this process are fewer branches, developers performing tests, more atomic continuous integration, and the use of mocking frameworks and tools. In essence, developers are continuously running experiments as in a scientific process model rather than the more familiar engineering process model.

Jeffrey also believes that the key to success with modern software development is achieving Continuous Delivery, and Forrester –in partnership with ThoughtWorks – developed a Maturity Assessment Model for Continuous Delivery with these five levels:

Level	Focus	Characteristics	Results
5	Hypothesis-driven delivery	Requirements include testable metrics Frequent use of A/B testing Services designed for CD DBMS changed decoupled from system changes	Delivery enables business innovation
4	Release on demand	Teams organized around services Deployment pipeline rejects bad changes Work delivered in small batches Comprehensive test + release automation	Service always in a releasable state Capability >= Need
3	Regular releases w/ milestones	CI and trunk-based development Automating provisioning and testing "Done" = tested and deployed	Regular release cadence Capability < Need
2	Time-boxed releases	Clear product ownership Change management controls <1 mo. cycles Some testing, release automation	Planned releases Capability < Need
1	Heroic individuals	Manual testing Integration explosion Manual provisioning	Ad-hoc releases

Figure 6: Continuous Delivery Maturity Model

Summary

Like the old adage that the only certainty is "death and taxes", ALM Connect and its Executive Day counterpart made clear that – while there are many techniques and tools available to the development community – the forces that face everyone include "complexity and speed". The most successful businesses will be those that best manage these many options to find the optimum combination of practices and tools that allow them to build – before their competition - the software that their users want.

Keith Pleas is the Organizer and Conference Chair for the ALM Summit. Keith is also a software architect, and has worked with the Microsoft patterns & practices, .NET Framework, and Visual Studio teams. Keith is an internationally known writer, speaker, conference organizer. Keith was a founding board member of INETA and created the INETA Speakers Bureau.

Microsoft ALM Certification – Part 2 of 3

Anthony Borton
ALM Trainer & Consultant
Enhance ALM

Welcome to the second of my three part series on Microsoft's new MCSD – Application Lifecycle Management certification. In the first part of this series I provided information about the certification as well as some guidance on how to prepare for the 70-496 exam.

In this article, we'll move on to how to prepare for the 70-497 exam. NOTE: You do not need to complete the exams in a specific order. I'd suggest you start with the one you're most confident on and then move through the remaining two.

Exam 70-497: Software Testing with Visual Studio 2012

This is Microsoft's first exam focusing primarily on Microsoft Test Manager (MTM) 2012. While this is the second version of MTM, it is the first time there has been an exam for it.

You can find the specific details of the exam on Microsoft's exam page for 70-497.

The audience identified for this exam on the exam page is "Developers". I would question this based on the *skills being measured* section of the exam page. I suggest that this exam is more directed at full time manual testers more so than developers.

Exam Details

Exam code:	70-497
Exam title:	Software Testing with Visual Studio 2012
Release date:	September 25th, 2012
Pass mark:	700
Questions:	Approximately 45
Time:	Approximately 145 minutes

Exam Preparation Best Bets

Finding a single fast-path to help you prepare for this exam is difficult. This exam focuses on your hands-on knowledge of how to use Microsoft Test Manager. Here's a list of three options for you to consider.

Practical	One of the best things you can do to prepare for this exam is to use Microsoft Test Manager and become familiar with all of it's features and capabilities.
eBook	Testing for Continuous Delivery with Visual Studio 2012 from the Microsoft Patterns and Practices group.
Formal training	If you want to get up to speed quickly, may be attending a formal training course might be an option. Take a closer look at this course if this is a path you might consider.

Exam Preparation – a closer look

The 70-497 exam comprised three key topic areas. As I did in part 1 of this series, lets look at each of these areas and try to find some helpful information in the MSDN library to help you study.

Before we look into each of the areas, one thing I will point out is that the majority of the exam focuses on the planning and management of your testing effort. Only 35% focuses on actually executing your tests. Remember this and apportion your study accordingly.

Another tip I will share with you is that at the time of writing this article, both TFS 2012 Update 1 and Update 2 have been released and Update 3 RC1 is available. The first two updates introduced a number of big improvements and new features to Microsoft Test Manager. Eg. the web-based Test Hub, cloning test plans and more. The exam has been written to match the RTM version of the product so you need to "forget" any features added in post release updates.

Create and Configure Test Plans (31%)

This topic covers the following high level areas:

- Create test plan properties
- Configure test settings
- Define configurations
- Create Test Suites
- Configure Test Suites

Much of the information for this topic can be found on the MSDN Library under a topic titled "Defining a Test Plan" (http://examcr.am/14CCryh). Make sure you read all child nodes underneath this topic on MSDN.

Manage Test Cases (34%)

This topic covers the following high level areas:

- Create Test Cases
- Create Test Steps
- Define parameters
- Manage Shared Steps
- Manage requirements

The MSDN Library link for this exam topic is titled "Creating and Managing Tests" (http://examcr.am/UD8bNY). Remember to read all child nodes underneath this topic on MSDN.

Manage Test Execution (35%)

This topic covers the following high level areas:

- Run Tests
- Perform Exploratory Testing
- Manage bugs
- Use Lab Center
- Analyze Recommended Tests
- Perform analysis
- Manage Work Items

The third and final exam topic area has two primary locations on MSDN for you to focus your study. Firstly look at "Running Tests in Microsoft Test Manager" (http://examcr.am/12DgK0W) and then "Performing Exploratory Testing" (http://examcr.am/10Dh0wW).

Good all-purpose resources

The following three books represent great resources, not just for the exams but for anyone working with or supporting Microsoft Team Foundation Server.

	Testing for Continuous Delivery with Visual Studio 2012 (Free eBook) (Physical book on Amazon) Authors: Larry Brader, Howie Hilliker , Alan Cameron Wills
	Professional Team Foundation Server 2012 Authors: Ed Blankenship, Martin Woodward, Grant Holliday and Brian Keller.
	Professional Application Lifecycle Management with Visual Studio 2012 Authors: Mickey Gousset, Brian Keller and Martin Woodward.

That concludes part 2 of this series. In my final post in this series, I'll provide guidance on exam 70-498: Delivering Continuous Value with Visual Studio 2012 Application Lifecycle Management.

Anthony Borton is an ALM trainer and consultant for Enhance ALM, an Australian based company specializing in helping organizations successfully adopt ALM best practices. He is the author of eight TFS 2012 training courses and delivers the most comprehensive range of TFS courses in the world today. You can find out more about these courses in the United States at www.quicklearn.com.

Release Management with Team Foundation Server 2012

Martin Hinshelwood
ALM Consultant
Northwest Cadence

While on the surface it looks like TFS 2012 has little in the way of support for release management, you would be wrong. There are many features in Team Foundation Server, many of them added in TFS 2010 that can aid you in creating a release management strategy.

A professional Development Team is supposed to create potentially shippable quality output that has no further work required for delivery.

There are many facets to release management and contrary to popular belief not a single one of them revolves around branching or code. There, I said it, its not about the code. Its about what you deploy. You may have a passing resemblance between code branching and your release process but they are two independent and only occasionally related processes. Intertwining them is where many organizations start to see friction, it is however avoidable.

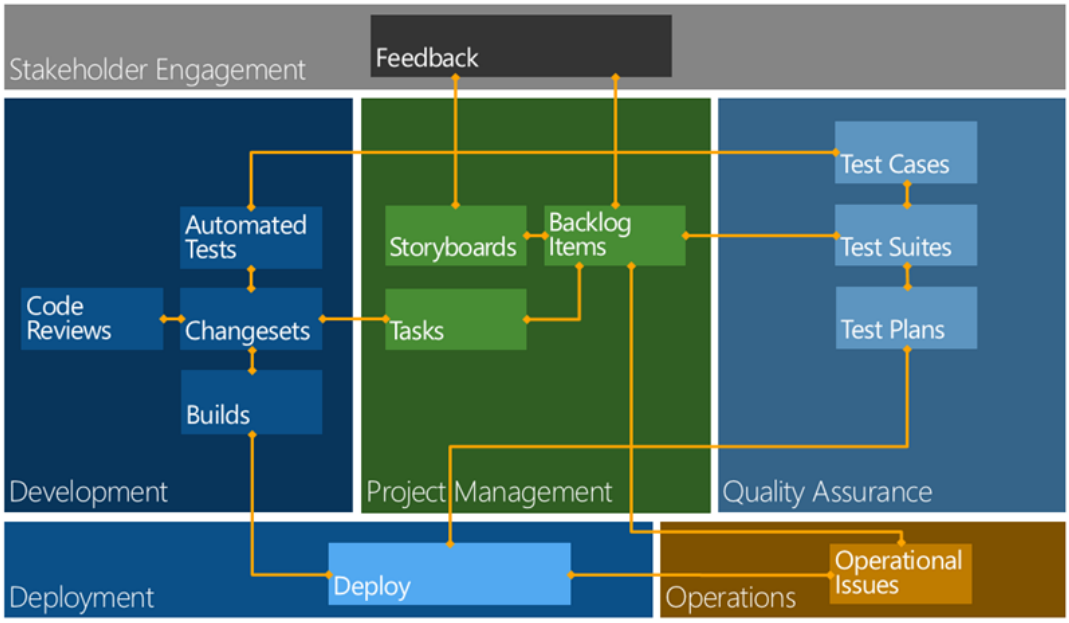


Figure: Workflow and relationships in Team Foundation Server 2012

Above I have a rough workflow for working with Team Foundation Server 2012. In this model you would create a ‘build’ from source control and the output would be ‘dropped’ to a well known location.

I have builds that just output binaries to network shares and I have builds that create Chocolatey packages. I even have builds that continuously deliver the output to production…

Figure: Automated Build in Team Foundation Server 2012

So now that I have a uniquely identifiable "build" I now need some way to repeatedly and consistently promote it through my environments. This is about deployment and we need to make sure that we deploy the same artifacts, hence the unique build, through the same process. Why, well so that we can be sure that not only does our application work, but that our deployment does as well.

In the case of most customers a hap hazard mish mash of technologies is then used by an overworked and under manned Configuration Management team, or worse Operations is just left of their own, to deploy that application to a variety of environments through to production.
Not only is this unnecessary for modern teams delivering modern software it is wholly unprofessional!

If you have people doing repetitive and mundane manual tasks then you not only doing them a disservice, but you are injecting unnecessary risk into the process. Automate everything and use your knowledge workers effectively!

The first thing that we need to consider for release management is some sort of delivery interface.

You, or your configuration management team, are not going to be able to really understand all of the tens and maybe thousands of applications that you will be asked to deploy. And nor should they; Microsoft does not ask us to understand the complexities of Office, they have an auto installer.

We need some rules around which our teams can create a concrete interface that shields teams beyond the development organization from the complexities of delivery. We have Engineering, Configuration Management, DevOps and likely Operations that all need to consume this interface. We could come up with our own, but there are well defined interfaces out there that were specifically designed around solving some of these, or similar, problems.
Creating 'installers' has been the tool of choice from yesteryear but they are wildly clunky and unwieldy in modern software development. I for one have always hated them…

So what to use instead?

You may have been using a little technology called NuGet recently. As of Visual Studio 2012 it ships out of the box and is a packaging and deployment technology for adding references to projects within Visual Studio. It has many satellite implementations that are based on that technology, Chocolatey for one, and you can implement it however you like. What is it? Well, it is fundamentally a Zip file with a Manifest, a Script and some files. It also takes care of dependency management.

There is a learning curve… its not all sweetness and light and there is no silver bullet. You still need to get the development teams to create and maintain the package and to fix it when things go wrong. But make no mistake… this **is** the responsibility of a processional development team.

A professional Development Team is supposed to create potentially shippable quality output that has no further work required for delivery.

So lets assume that we have some quality output and that output includes the scripts necessary for deployment, ideally contained within a NuGet package. Even in that happy state we need to take that output and get it to an environment, deploy it and potentially tests it.

Lab Management for Release Management

For delivery to environments owned by your engineering teams you can use Lab Management to deploy to almost any sort of environment that you want. Remembering that just because I say 'engineering teams' does not mean that there are not representatives from operations on those teams, because there should be.

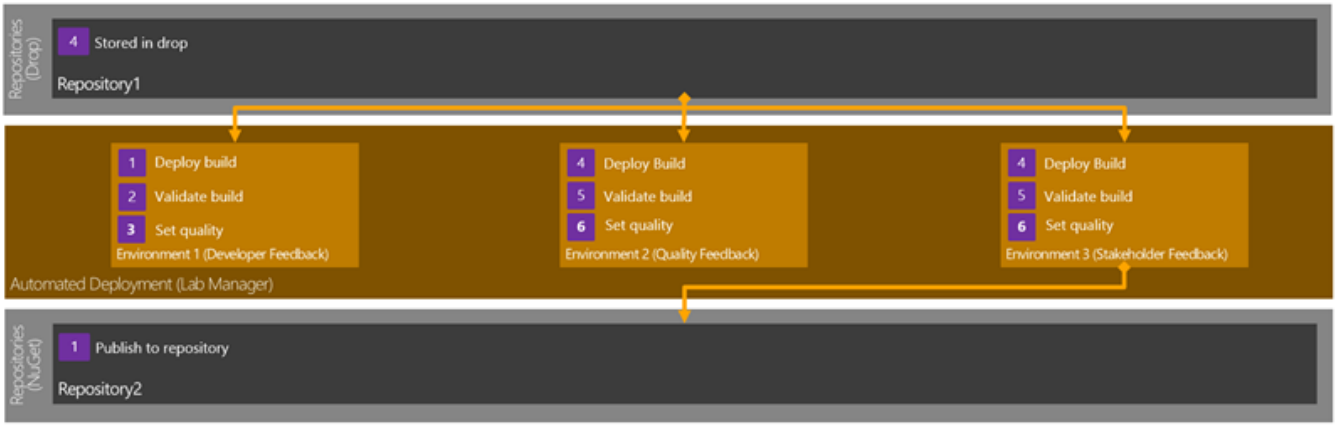


Figure: Release Management with Lab Management for Feedback

Note This can be used to deploy all the way to production but it tends not to be. Operations frown upon installing development tools in production for some unknown reason.

You can create Standard Environments or use System Centre Virtual Machine Manager (SCVMM) dynamic environments that are constructed from one or multiple machines into a single identifiable thing that we can deploy to.

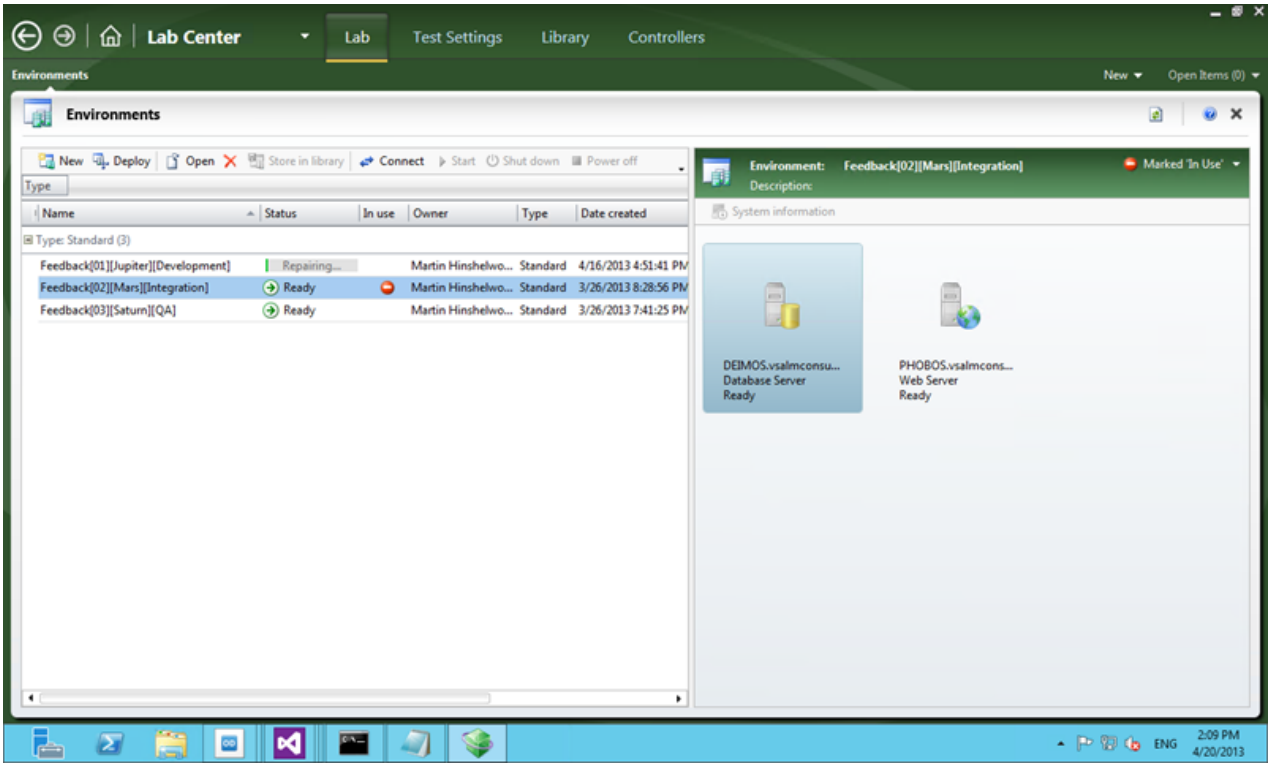


Figure: Environments in Lab Center for Release Management

Lab Management leverages Team Foundation Build (TFB) and allows you to not only see each of the machines within an environment but also to deploy a pre-compiled build to that environment and orchestrate running of your integration tests against it. This is fantastically powerful as with no customization we can have a scheduled and automatic deployment of our software to an environment and then have a chosen set of automated tests run against it.

Note Team Foundation Build (TFB) is not just for compile and test. It is built around Windows Workflow and can do pretty much anything your want across one or many machines.

In addition if we are able to spring for a System Centre Virtual Machine Manager (SCVMM) environment you can have your environments automatically spun-up from template environments and snapshots created prior to the deployment and test. And all that is still out of the box functionality.

You could add some capabilities with some simple customization of the build workflow to for example; deploy and verify an environment for each of our 20 testers. The capabilities are endless…

Note You do not need Hyper-V or SCVMM to take advantage of Standard Environments. You can use VMWare or even bare metal.

So if we are building, deploying, verifying and testing within our engineering teams then we would gain the most capabilities from using Lab Management. Lab Management is likely part of your existing licensing (you just need > [Test Professional | Premium | Ultimate]) and can be used at no additional cost. You can setup and configure 'standard environments' in minutes to start taking advantage of this now…

Octopus for Release Management

Remember I mentioned earlier that your operations teams would be… uncooperative… if you wanted to use Lab Management deploy

to production? While I have some customers that are perfectly happy to have a Build Agent or Test Agent installed on their servers it does feel a little… dirty. We really want a clean orchestration engine that is much less invasive and our operations will accept.

At this stage we already have a 'package', probably a NuGet package, that has been deployed and promoted by the engineering teams through their verification paths and they have marked certain 'builds' as 'done'. Those 'done' builds have likely, if they are NuGet packages, been deployed to a repository that we can easily call for our operational deployments.

Figure: Release Management with Octopus for Production

The reason that I am again mentioning NuGet again is that there is an application called Octopus that has been specifically designed to deploy NuGet packages with ease. Make no mistake, this is an operations tool that has been designed from the ground up to support deploying the output from professional Development Teams to production.

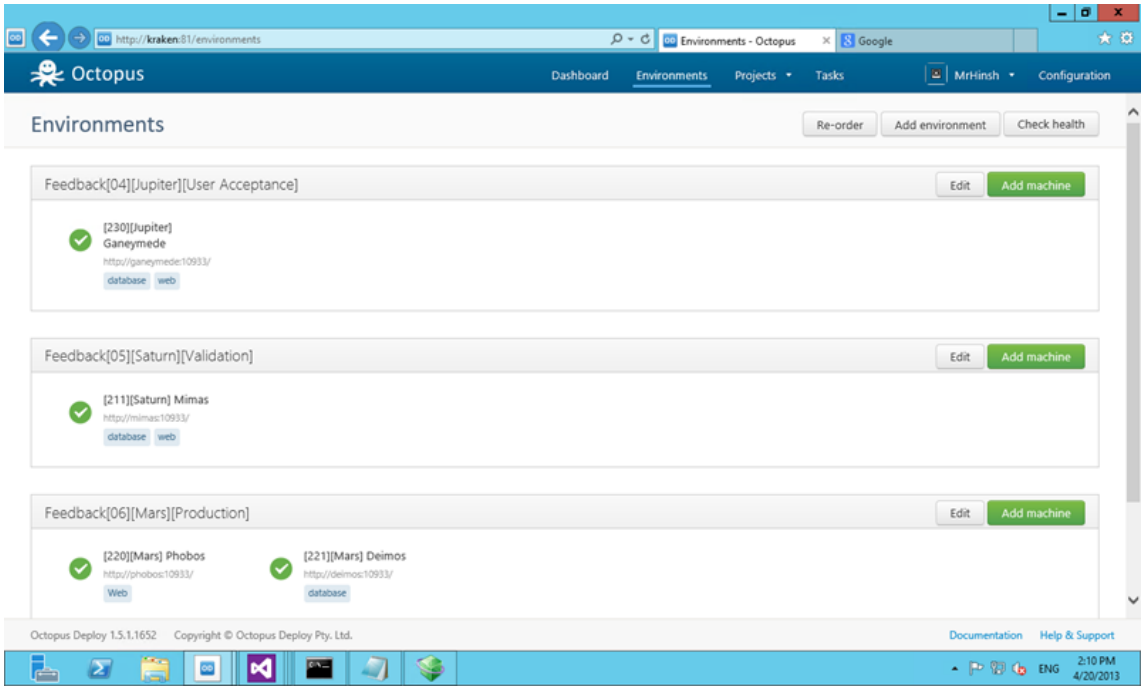


Figure: Octopus for Operations

You create Environments and then Releases and you can have those releases promoted through those Environments. You can control permissions on who can promote releases through which environments and generally get and idea of what is deployed where.

If you are using NuGet packages you will even know what is deployed.

Conclusion

There are lots of options for the automation of your application deployment. I can't say this enough but:

A professional Development Team is supposed to create potentially shippable quality output that has no further work required for delivery.

issues so that you don't get many, if any, issues in production. Teams that follow models like this, when combined with good engineering practices, spend less time fighting fires and more time delivering real customer value. The goal would be to get from Check-In to your Production Environment at least every 30 days.

That means that you need to be using the very best Processes, Practices and Tools to make it as slick as possible.

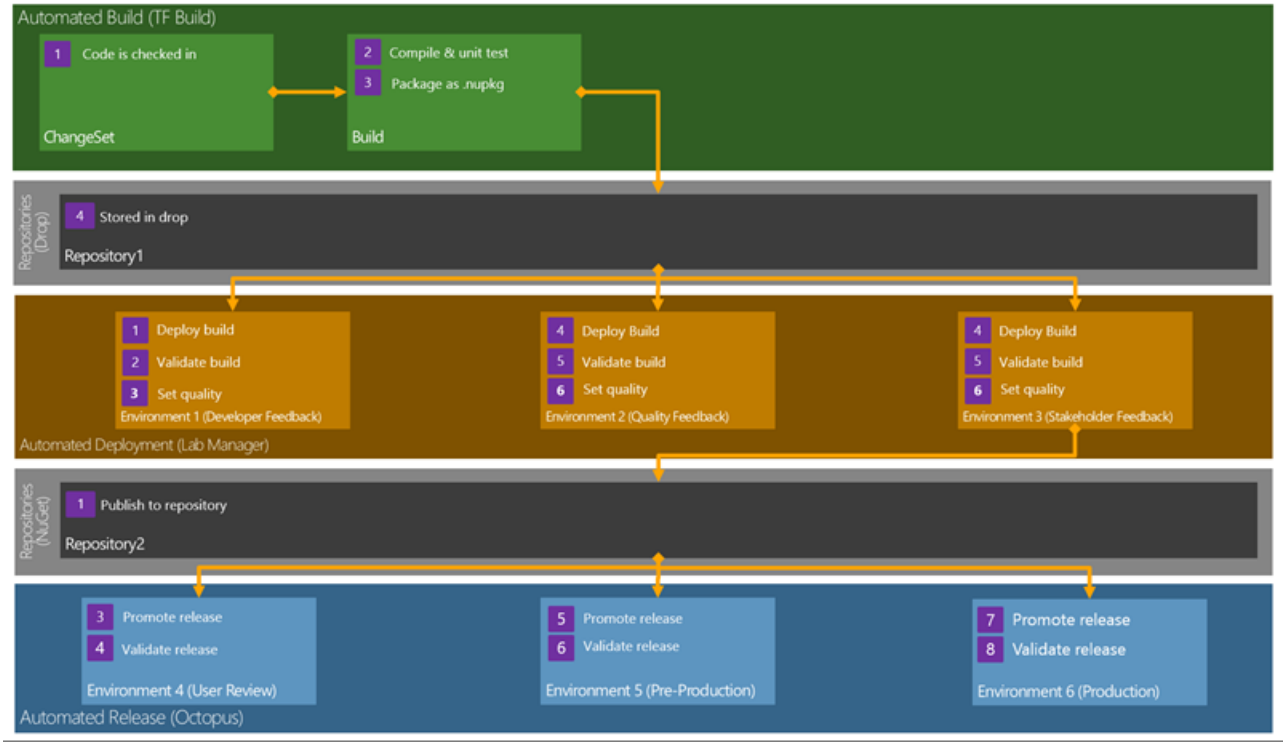


Figure: Release Management Workflow in Team Foundation Server 2012

If we fail any sort of stage gate that occurs after the engineering team has said that it is 'done' then we need to seriously look at why that happened and what we can put in place to make sure that it never happens again. That sort of attitude will improve the quality of your software and reduce the risk of delivery vastly.

This flow of building once and then repeated validation will help weed out those last quality

How long is your release process?

How sure are you of your quality?

Martin is an ALM Consultant at Northwest Cadence, and Visual Studio ALM MVP (2013). He was named ALM MVP of the year 2011, ALM Ranger Champion for 2011 and was one of the first Professional Scrum Trainers with Scrum.org. Martin regularly delivers consulting and training while writing for his Visual Studio ALM blog, and is a sought-after speaker.

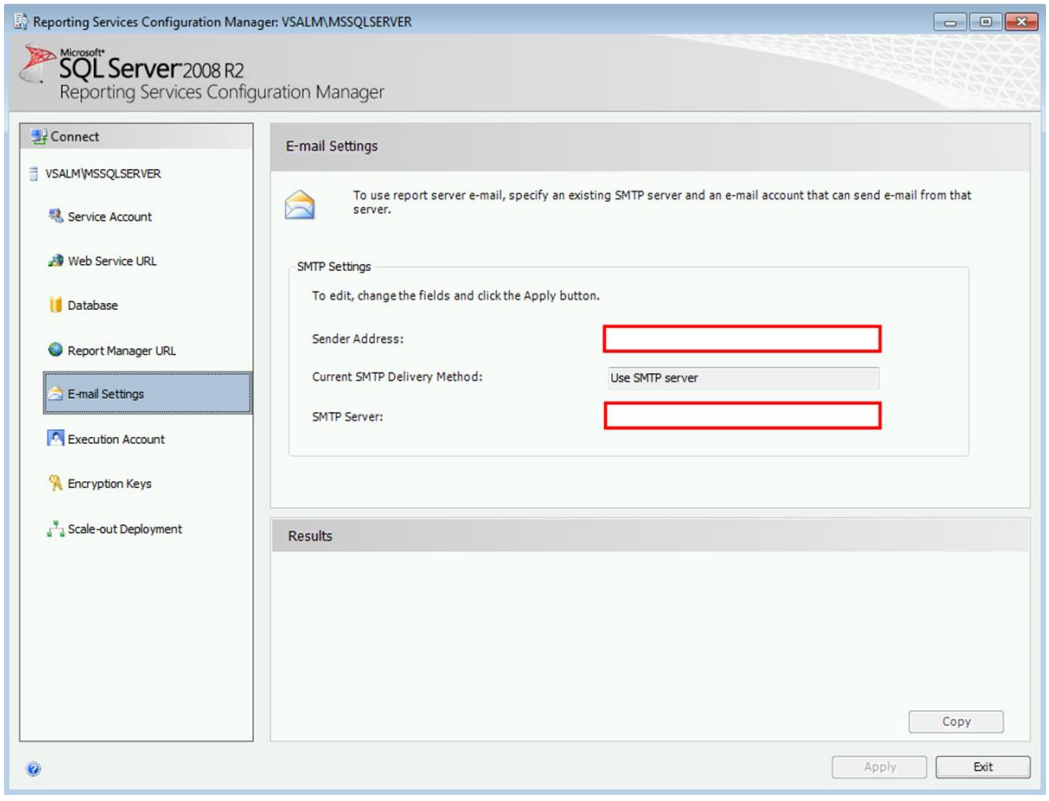
TFS Admin Reports delivered to your inbox daily

Anna Russo
ALM Consultant
Imaginet

I came across a TFS cube error while troubleshooting another issue that was reported to me. Now if every TFS Admin had the TFS Admin Reports emailed to them daily, then they would have seen the cube status error and responded accordingly.

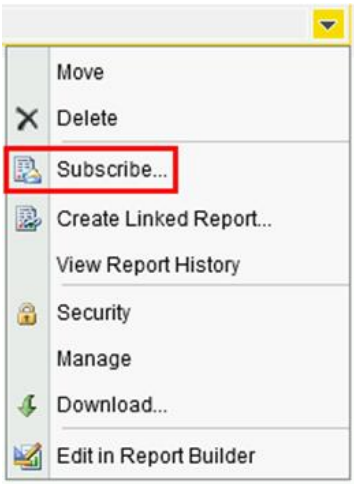
It's really easy to setup Reporting Services to automatically email you the TFS Admin reports so that you are proactively managing the TFS server.

1. Navigate to Microsoft SQL Server > Configuration Tools > Reporting Services Configuration Manager
2. Fill out the Sender Address and SMTP Server

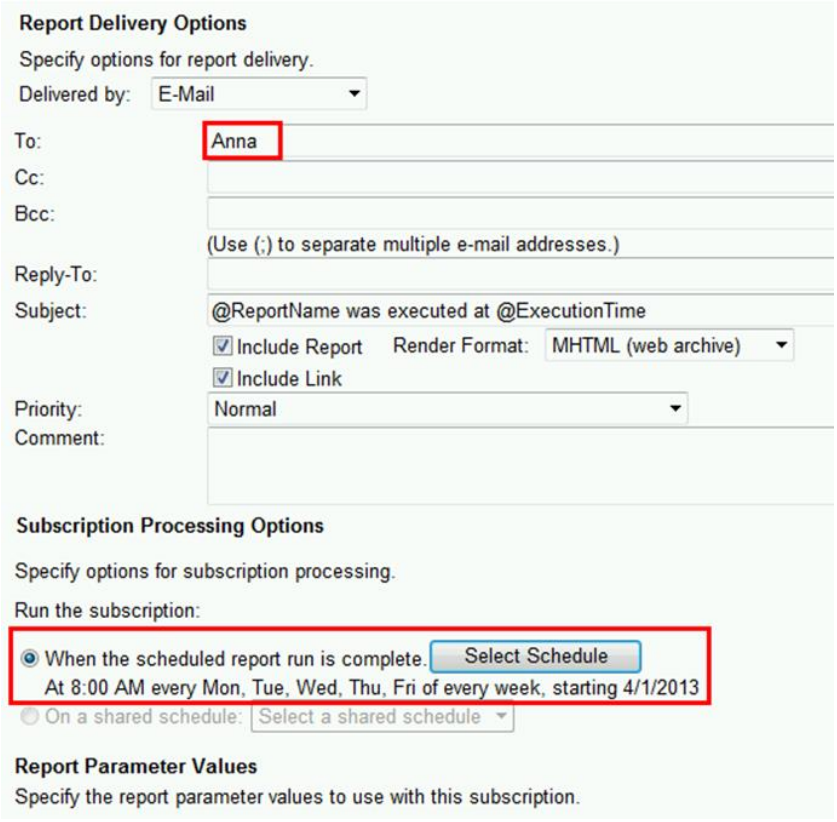


EDITOR'S NOTE! For additional information on TFS Admin Reports click on the following link for an MSDN article by Grant Holliday
http://blogs.msdn.com/b/granth/archive/2010/07/12/administrative-report-pack-for-team-foundation-server-2010.aspx

3. Go to the each one of the TFS Admin Reports and select Subscribe



4. Enter your email address and time of day you would like to have your report emailed.



The TFS Admin Reports will be emailed to you (now if you set up an Outlook rule and ignore these emails, it kinda defeats the purpose of proactively managing the TFS server, just saying)

BTW, this also works for any of your other TFS SSRS reports that you would like automatically emailed out.

Anna Russo is an ALM Consultant and MVP with Imaginet, a firm that specializes in helping companies improve software development process with TFS. She has worked with a variety of clients and upgraded/installed TFS, provided training, created custom processes, implemented release strategies, and automated build scripts. Visit Anna's blog ImprovingSoftwareQuality.com

The e-Magazine for:

ALM - Application Lifecycle Management

DevOps - Development & Operations

Agile - Scrum, Kanban, etc.

TFS - Team Foundation Server

Click for more Details...

ALM Mag Brings You Each Month:

- **Articles, Whitepapers, Case Studies and Videos and Audio** providing:
  - Advance notice and Previews of new versions,
  - **Reviews** of new versions and updates,
  - **How To** and **Step-by-Step Guides**,
  - **Tips** highlighting **Primary Tools** and
  - **3rd Party Integrated or Bridging Tools**,
  - **Processes, Books and Events**
- **Surveys** to discover more about the types of content that interest you so we can serve you better
- **Industry Study and Survey Results** to provide you more insight into what others in your community are doing
- **Interviews with top ALM Leaders** to help you discover what's coming and how you can get there
- **Tutorials and Training Courses** to increase your knowledge and leverage your career
- **Event and Conference Information** with a spotlight on Details, Agenda and Calendar so you are aware of what you want to attend to increase your career potential
- ...and finally **IT Humor** (yes it does exist) including **Cartoons**

and more...

This Magazine Is For You If You Are:

- a CEO, CIO, COO
- an Application or Development or QA or DBA Manager or Architect
- a Development or QA Lead
- a Product Owner
- a Project Manager
- a Scrum Master
- a Business Analyst
- a Tester
- a Developer
- ...Or anyone involved in software Planning, Development, Testing, Infrastructure Provisioning or Deployment

Subscribe

Index Page

What's New in TFS 2012? – Team Explorer

Mohamed Radwan
Senior ALM Consultant
Marvel ALM

This series introduces what's new in Visual Studio 2012 and Team Foundation Server 2012 (TFS 2012). This article focuses new features in Team Explorer.

There are many clients for TFS (Team Foundation Server) like, Microsoft Excel, Microsoft Project, Team Web Access, and several other clients including Team Explorer.

What is Team Explorer?

Team Explorer is considered the primary client for TFS. It is an add-in to Visual Studio 2012 or it can be installed as a separate standalone application.

There are a lot of changes, new and merged features in the new Team Explorer and Visual Studio 2012 e.g. the Pending Change Window merged within Team Explorer. Also there are new windows added to the Team Explorer, e.g. there is a new window called My Work, introduced in TFS 2012.

In this post I will focus on the Team Explorer itself. and its key features.

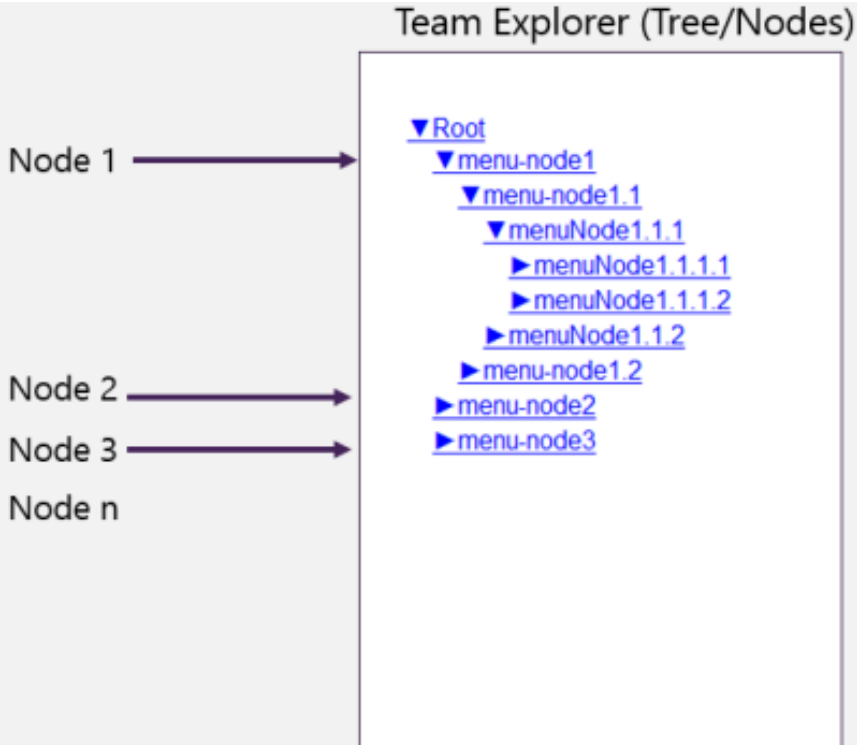
Team Explorer 2012 Enhancements

1. Architecture (Web Based)
    a. Navigation and Extensibility
    b. Work item search
2. New and Merged Windows
3. On Demand Data retrieval (paging)
4. Smart commands and Context driven
5. Reducing Modality
6. Performance and Async Operations
7. Rollback in the UI

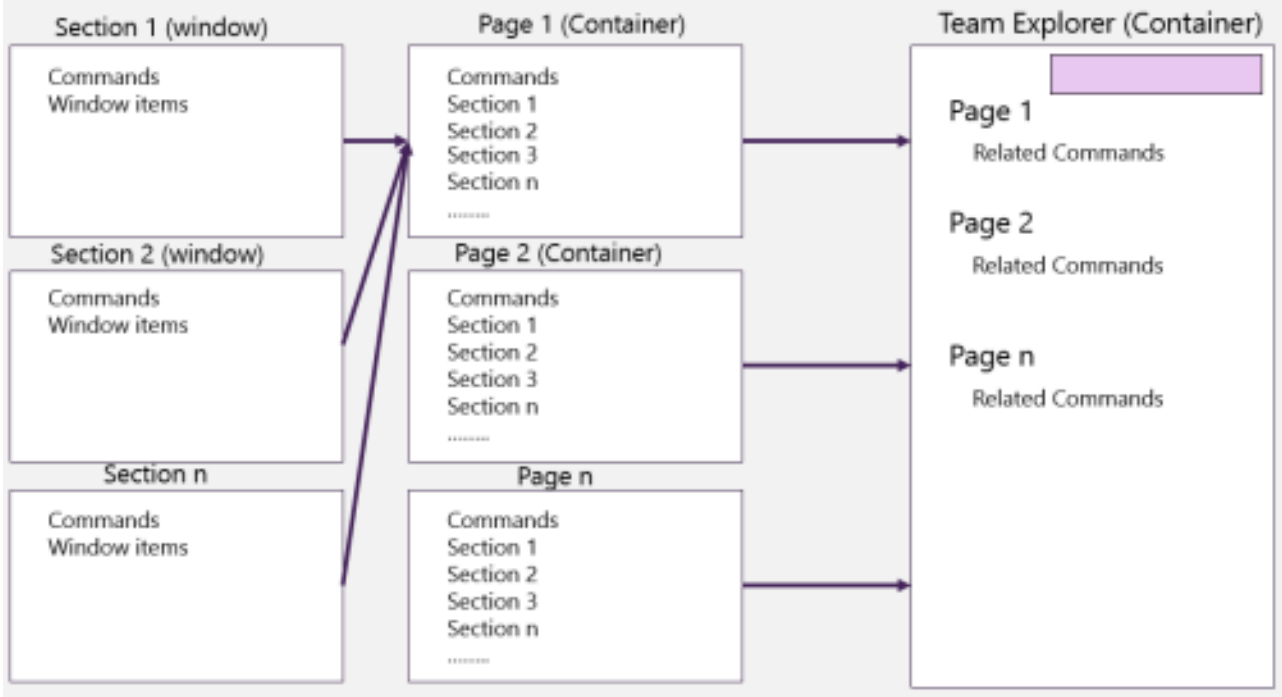
1. Architecture (Web Based)

a. Navigation and Extensibility

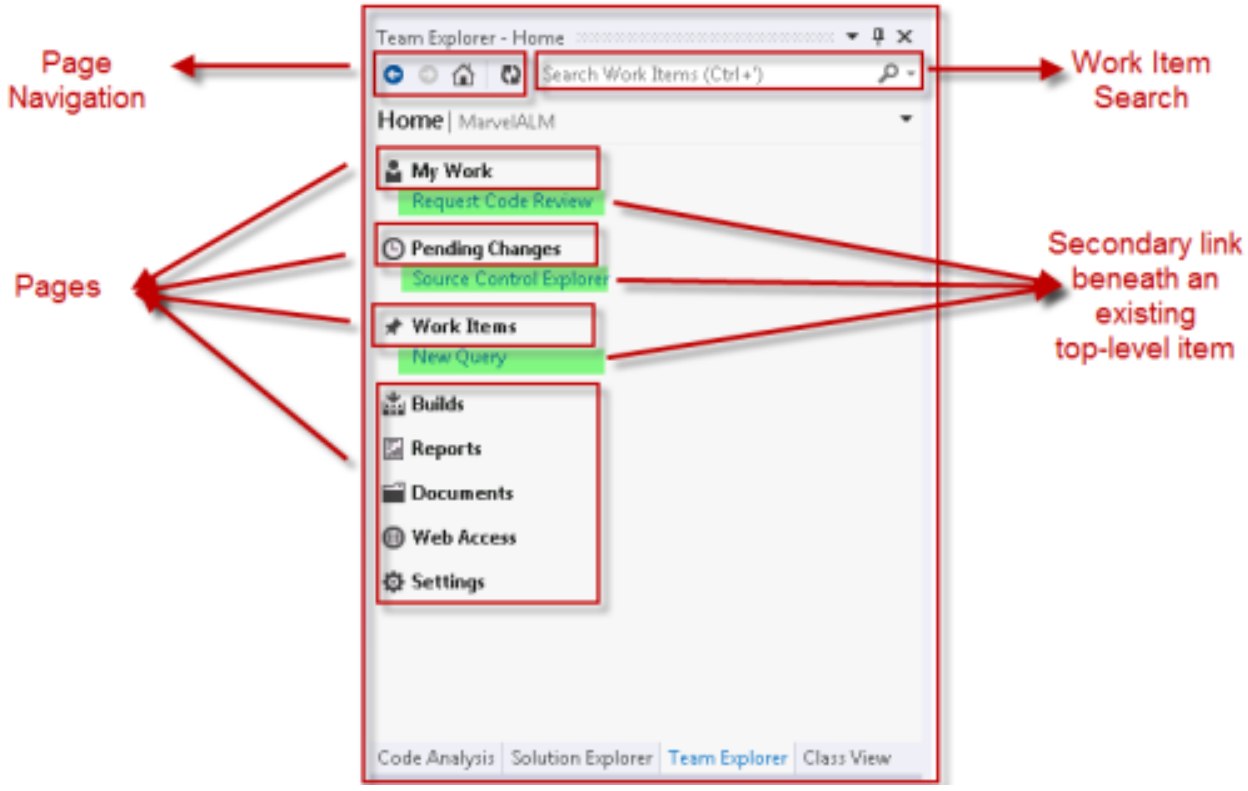
Old Team Explorer



New Team Explorer



The old version of Team Explorer was based on Tree/Nodes architecture which was very limited for new extensions, so whenever any extension or feature needed the only way to add it, is to add new node which makes the tree more complex than expected. In the new version, the architecture is based on web navigation style, the Team Explorer is like a main container that has multiple pages and each page is a container that has multiple sections and each section carry one window with many commands.



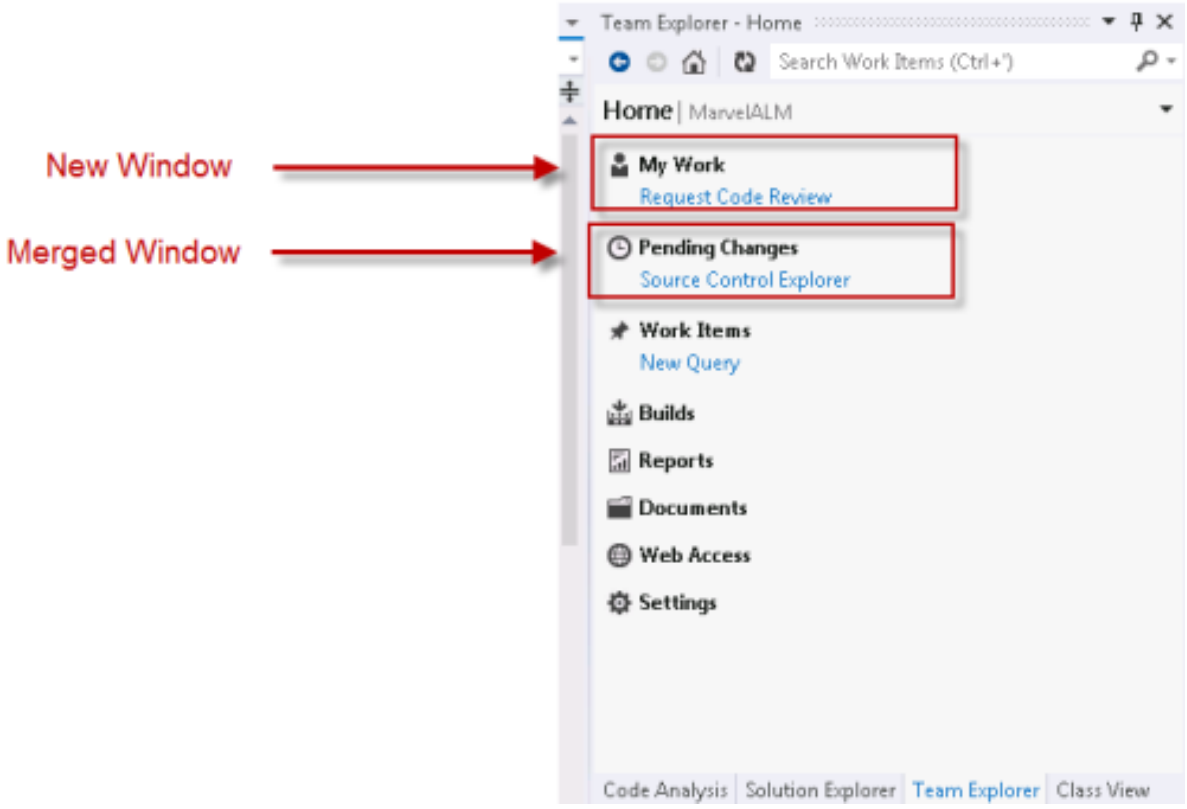
There is a navigation bar to go home (main container) at any point of time and also there are back and forward buttons to move between pages and sections as you visit them. This architecture gives Team Explorer richer extensible model, so we can extend it as the following methods:

- Add new page to the navigation structure
- Add a new section to an existing page
- Add a top-level link to the Home page
- Add a secondary link beneath an existing top-level item in the Home page

b. Work item search

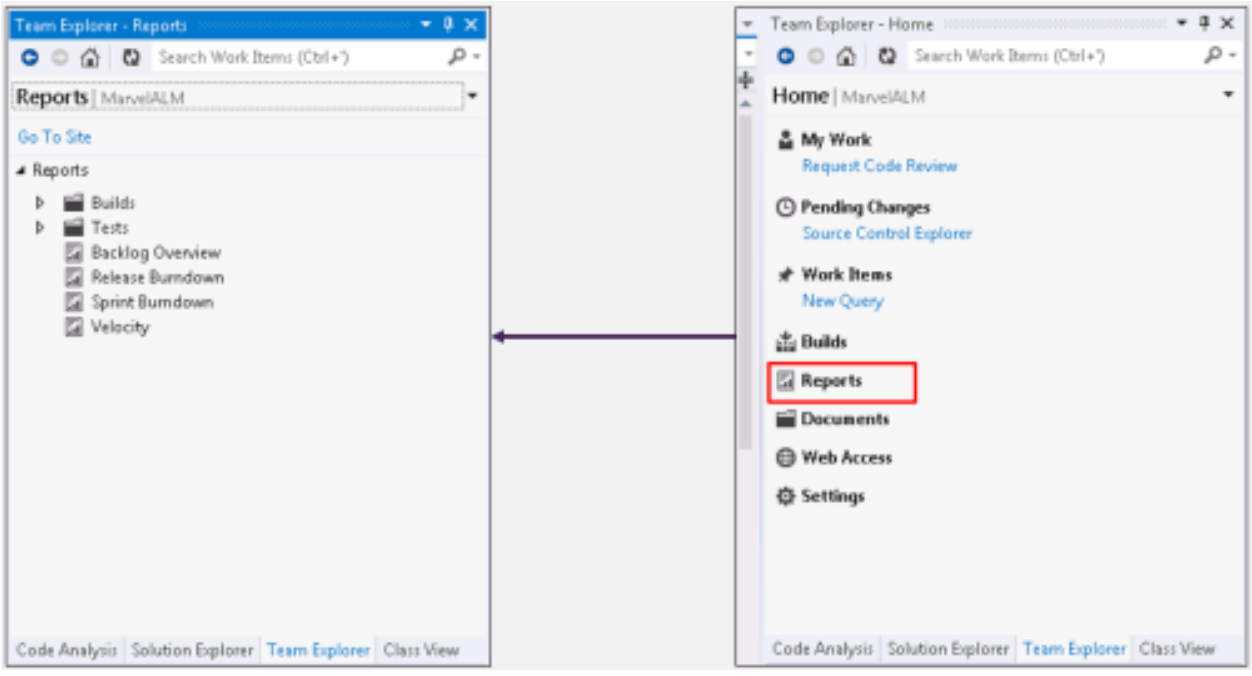
There is a work item search that we can search inside the work item no third party tool needed, we expect this feature to be more complex for different release of the Visual Studio and TFS, there are some search syntax that can be found here,[Work Item Search Syntax](#).

2. New and Merged Windows



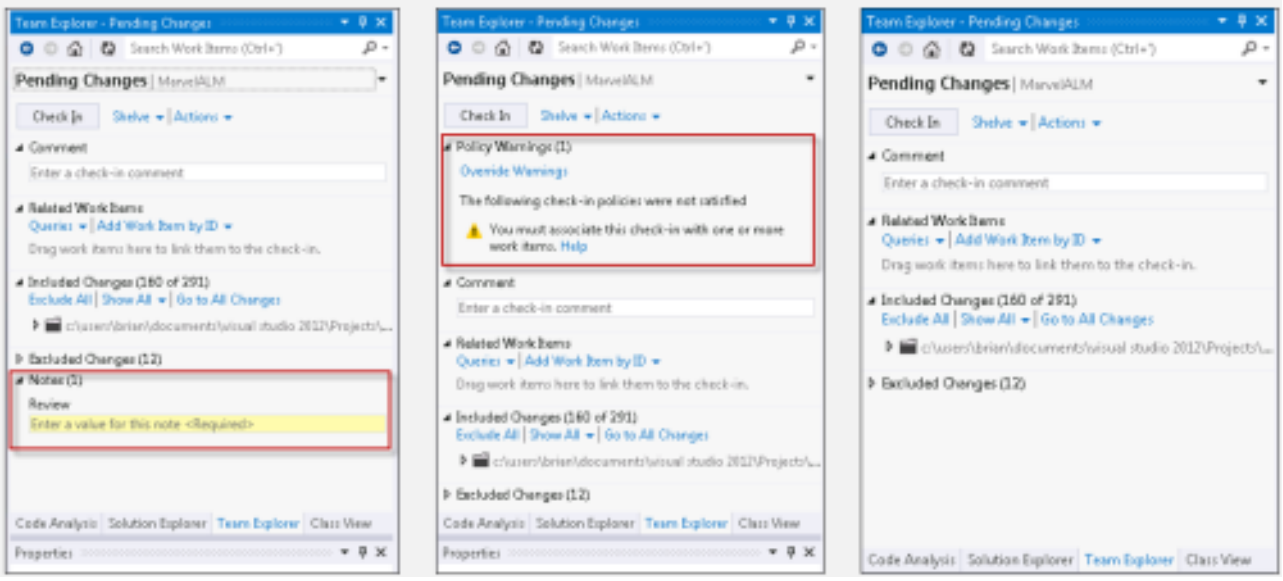
There are new windows introduced in TFS 2012 and new windows merged to the Team Explorer ex. My Work is a new window introduced in TFS 2012 and Pending Change Window merged in Team Explorer.

3. On Demand Data retrieval (paging)



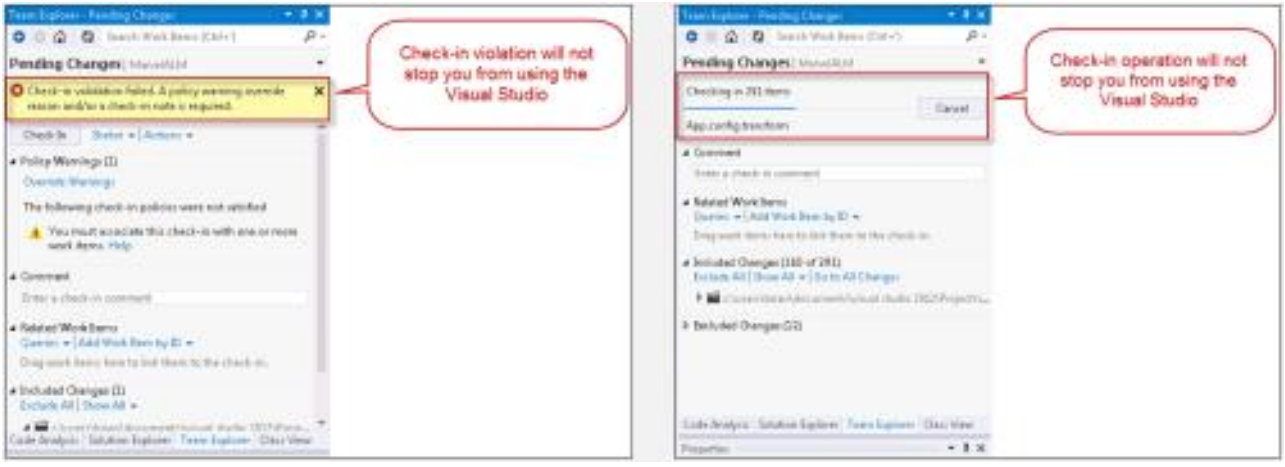
In the old version of Team Explorer when we open team project we had to wait for all files to be loaded and download like (Reports, SharePoint documents) and so on but in this version of Team Explorer noting will be loaded until you need it's like the paging concept when we displaying a lot of records, once we requesting the next page it will connect and bring us what we need.

4. Smart commands and Context driven



There are a lot reducing of right click and choose commands by exposing them and being intelligent, ex. the check in policy violation section will not show up if we don't have violation but if we have? It will show up and appear, so there are a lot of command will not appear if there is no need for them.

5. Reducing Modality



One of the things that we really hate are modal dialog boxes, they show up and don't let us do anything while they are there. There are a lot of improvements to reduce modality as much as possible. The check in pending change will not be modal anymore, so we don't need to wait for all our files to be checked in so we can start to use our Visual Studio, so now we can use the Visual Studio while the code checked in.

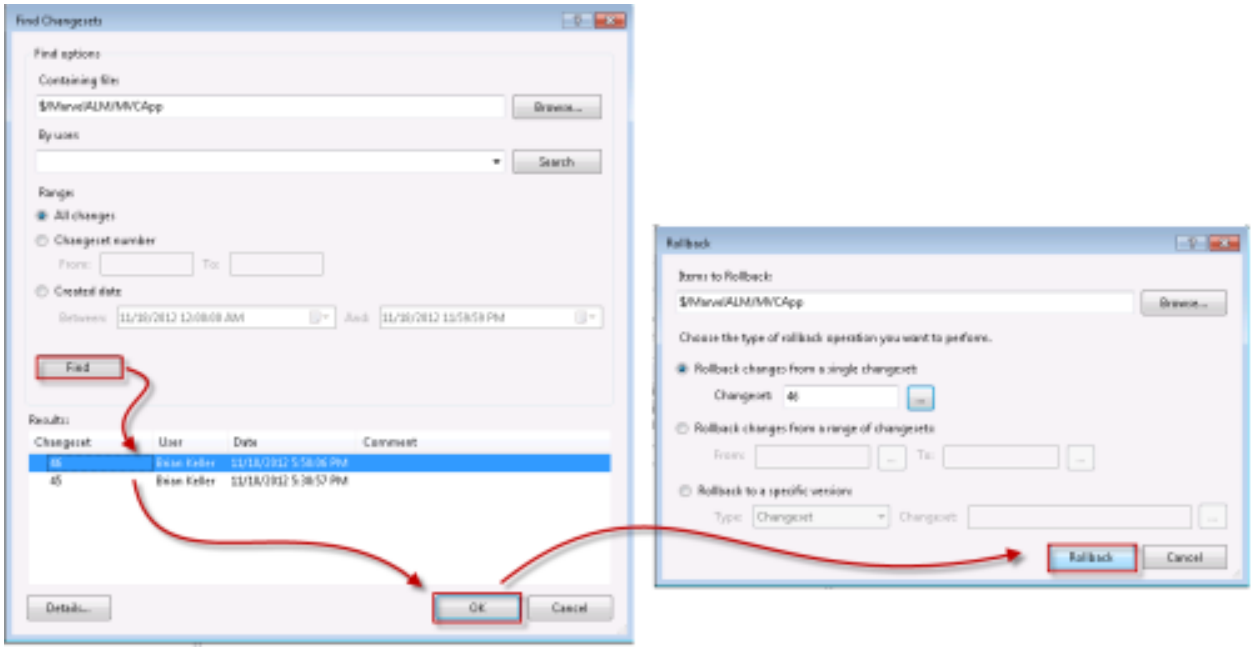
6. Performance and Async Operations

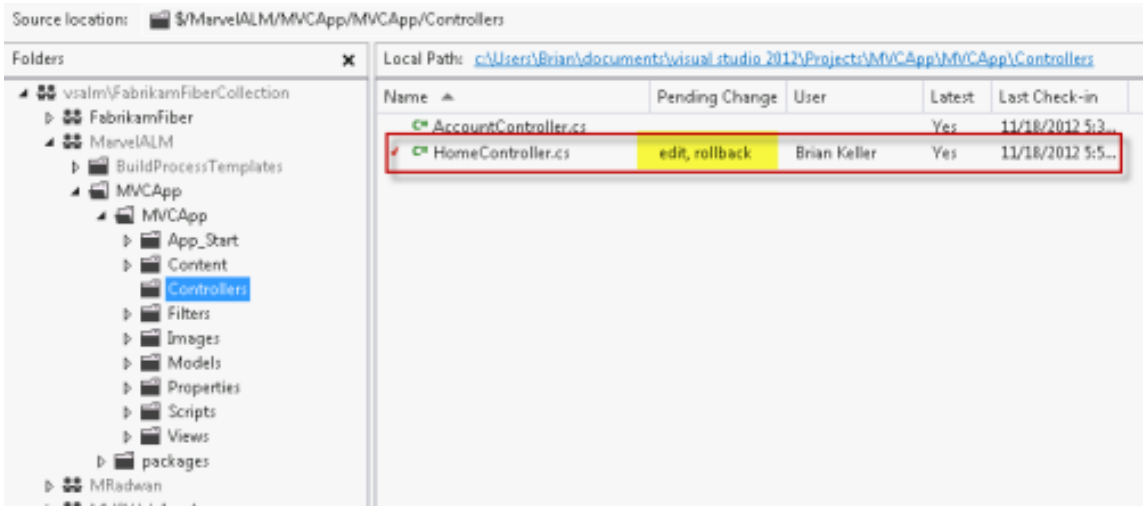
To enhance the performance, Microsoft worked hard to move long-running tasks to background threads wherever possible, also there are a lot of improvement of operations response so we can achieve responsiveness UI, this made by increasing the number of Async Operations that interact with the TFS with the Reducing Modality feature which introduce a very responsive UI.

Old and already Async Operations:

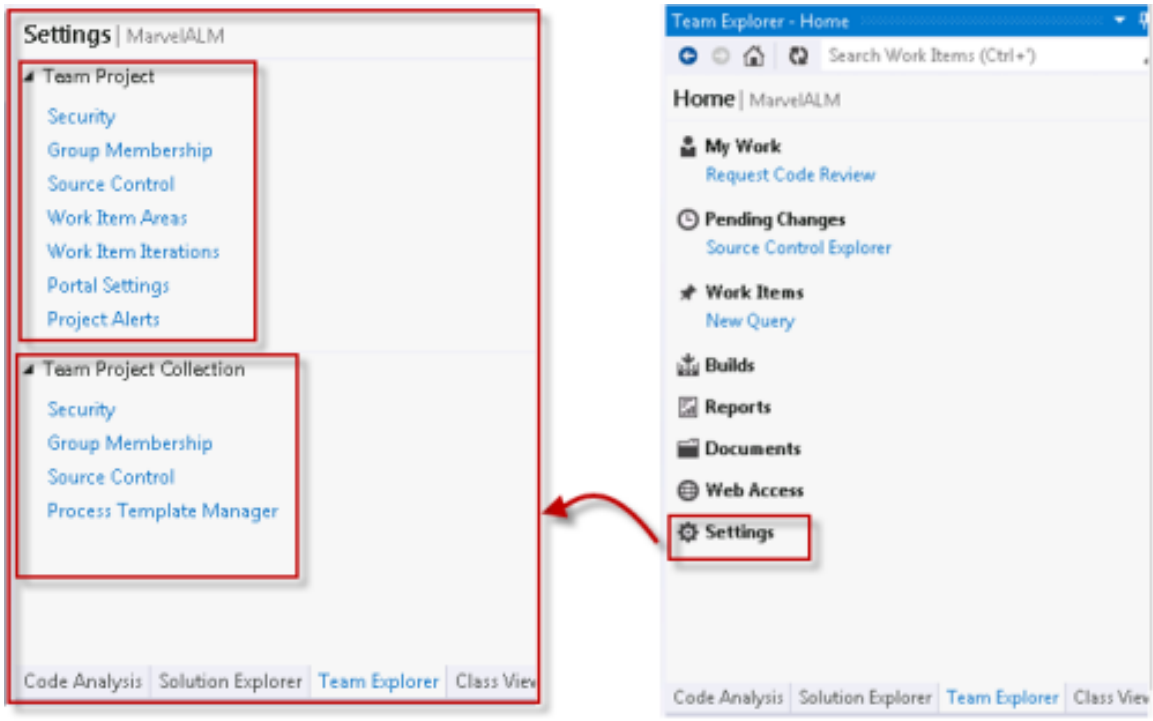
- History
- Annotate
- Source control explorer
- New Async Operations:
- Checking in
- Editing a file
- Find shelveset
- Sheveset details and changeset details
- File compare
- Open the work item

7. Rollback in the UI





It's not part of the Power Tool as a third part library anymore, it now become in the box. The project settings now also looks different.



Summary:
Team Explorer one of the major change of the TFS 2012 and there are a lot of new features and enhancements that can't be cover in one or two posts, as a summary, Team Explorer has different experience with a lot of change so this post just highlight the main and significant changes of the new experience introduced in Team Explorer.

Mohamed Radwan is a Visual Studio ALM MVP, Senior ALM Consultant at Marvel ALM with more than 10 years of software industry experience, he focuses on C#, MVC, JQuery, BDD, TDD, TFS and Practical Agile Implementation. A founder of TFSEG User Group, co-founder of MEAALM Community, author of DevMagicFake Mocking Framework, Mohamed is a frequent blogger and speaker at Microsoft events.

Countering 6 typical arguments against an Agile/Lean approach

Tomáš Tureček
Lean/Agile Coach
Tieto Corporation

The ambiguous nature of software development and thus the need for a different contracting approach – Agile/Lean contract. Let's now take a look at what the **typical arguments** against it are and how do we react to them.

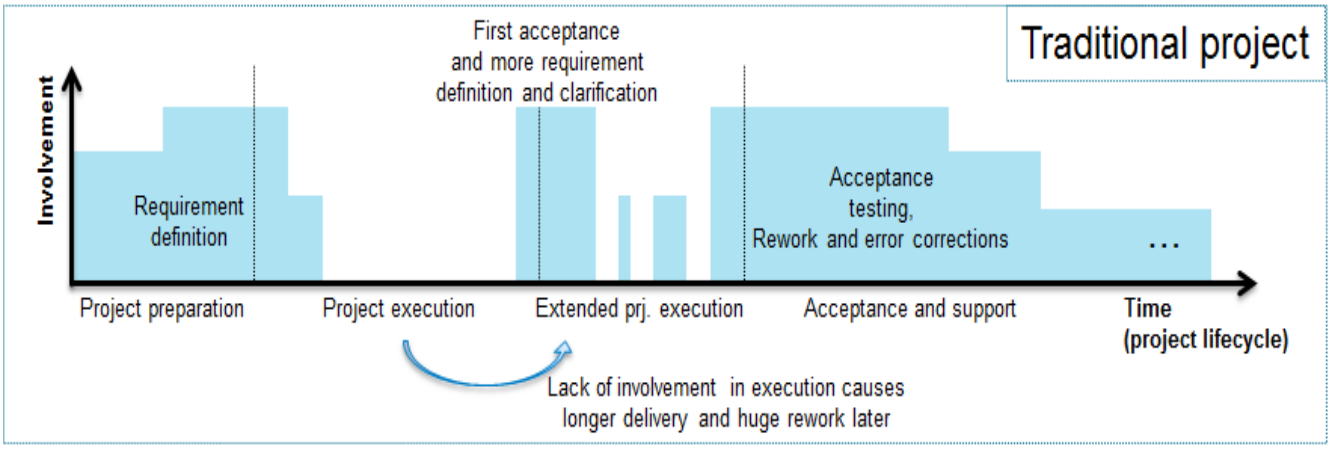
- **Time** – "We don't have time to be involved in the development too much"
- **Fixed price** – "We need to know the cost of the solution upfront"
- **Fixed scope** – "We need to know what we are buying"
- **Delivery date** – "How can you fix a delivery date if everything keeps on changing?"
- **Risks** – "You are trying to transfer the risks to us by that open scope model"
- **Tenders**

Agile and lean contracts solve one of the root causes of tension between customer and supplier as noted here in removing tension between customer and supplier.

1. Time

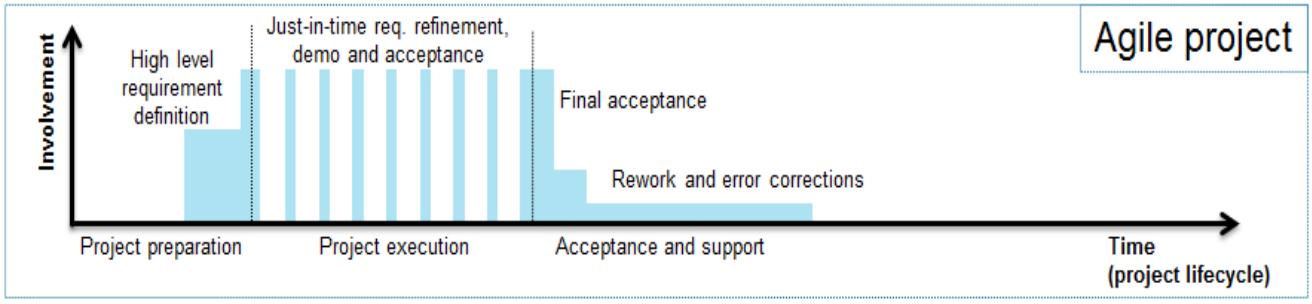
- We don't have time to be involved in the development so much"
- "Our people are very busy; they cannot attend demos and spend time with you on phone"

Well, what to say. Surely customers do not have time when they spend months in writing down very detailed requirements specifications at the beginning and then spend weeks or months on acceptance testing at the end to be sure that they got actually what they requested for. And after they find a result, they spend a few more months on rework, arguing and in error corrections. Whether you like it or not, this is the most common work flow of traditional projects; blue parts in the picture below emphasize work of customer people during a project.



Customer involvement in traditional project

But we can actually do everything in a much smarter way. Instead of writing down all detailed specifications at the beginning/upfront, we would rather define high level business objectives and understand needs together and perhaps contribute to the requirements definition. Then we should validate that we understand everything well through continuous delivery in iterations/Sprints. In the end – **customer time spent is less or equal** in comparison to traditional project. **And a result is acceptable at the first try.**



Customer involvement in Agile project

Another advantage might be that **involvement is more even and predictable, so it is actually very easy to plan upfront**. Like attendance on demos and iteration/Sprint preparations.

2. Fixed price

- "We need to know the cost of the solution upfront"

That is a very valid point. Sometimes we, as a supplier, do not realize that people from customer organization we talk to have their own limitations. Big companies usually have their internal IT department which works in budget planning mode. Like in one delivery I cooperated with some time ago:

- Customer: “We insist on fix-all contract. We cannot go with you for the open scope model.”
- Supplier: “Why?”
- Customer: “Because we need to know the price of future solution upfront.”
- Supplier: “Why?”
- Customer: “We need to ask for budget internally. This will enable us to order implementation of change requests from you.”

But for that we do not need to fix everything into a contract. We can **define maximum price** of the solution, so that customers can internally ask for a budget. And after that we can together figure out how to deliver the best possible value within this budget. Via iterations/Sprints customer will have 100% visibility about what’s going on inside a project and **together we can steer** it to **get the most out of time and the budget** left. All is usually much cheaper, without risk buffers and financial pillows to mitigate risk of commitment to uncertain requirements. And do it without rework during the acceptance period. In this way, the supplier can be more cost-effective, faster and more valuable to all stakeholders.

The case I mentioned was even easier because there was almost a decade’s history of working together, so we duplicated last year’s budget and started to discuss a cooperation model to get the most out of it.

3. Fixed scope

- “I need to know upfront what I am buying”

That says a lot about mutual trust. The more we are disappointed from previous cooperation (even with different business partners) the more we want to control and limit a future one. That actually leads to even worse results. **Death spiral.**

If this is the case, it is important to realize that we are inside a spiral and try to find the way out. I mean – do things smarter not harder. To the comment “I need to know what I am buying upfront” everyone should ask themselves *“In the past, did I get what I wanted in the end? No delays? Issues?”* Definitely not, otherwise there would not have been this trust problem. **Fixing detailed requirements upfront steals focus from *what is really needed* to *what is contracted*.**

During a discussion about new custom product I remember one lady from the customer’s representatives pointing her index finger at her coffee cup and saying: *“I need to know what I am buying – like this coffee. Its size, brand, flavour and price”*. Our colleague reacted to the comment: *“Is it really coffee that you need? What is the business objective? To get refreshed? Then we might find cheaper and healthier solutions. Like using guarana instead of coffee, more frequent breaks or just simply opening a window.”* This example opened the lady’s eyes and enabled us to continue further.

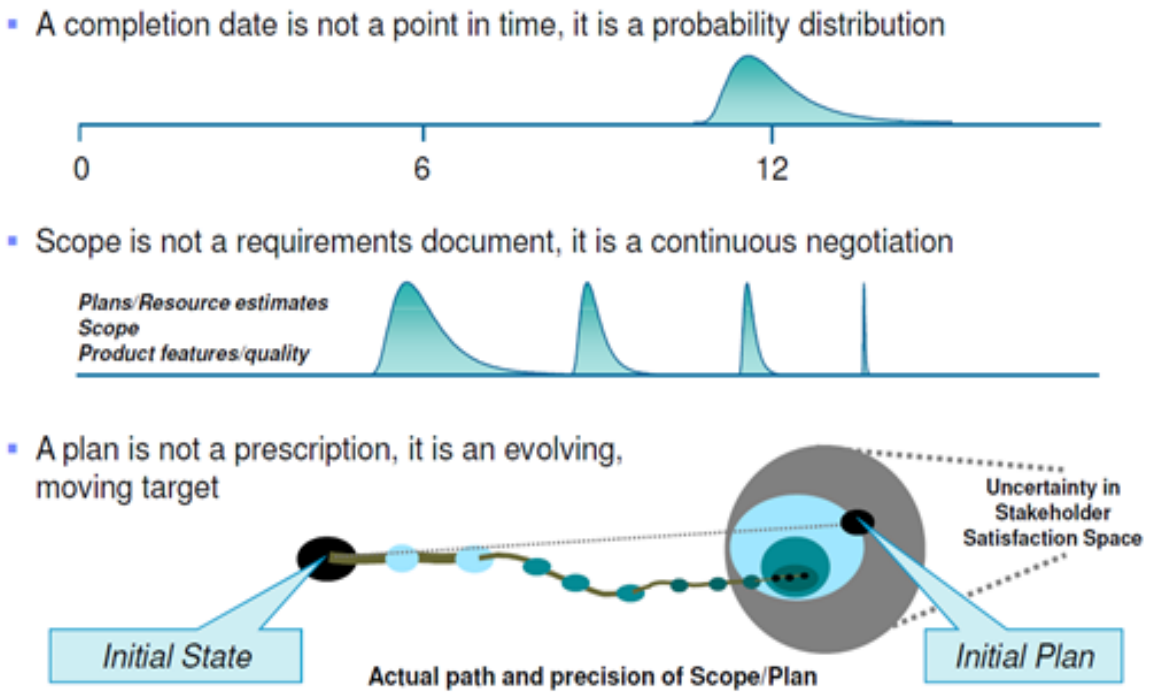
Building software is not the same as making repeatable products such as a cup of coffee. So applying the same principles in different contexts leads to using wrong tools, models and more disappointments.

4. Delivery date

- “I see you doing iteration/Sprint one after another. When you will be able to deliver?”
- ”Will these Sprints ever end?”

In other words; supplier produces functionality chunks in each iteration/Sprint. How can we say when it actually ends if we do not know all the detailed requirements upfront?

Let me share one picture (below) I have taken from Walker Royce presentation (Improving Software Economics, RSC 2009) where he says that the release of fulfilled needs is not a point in the timeline. It is more like a probability of delivering at some date since there are a lot of uncertainties in the beginning of the project. The more we work on the project, the more certain we are about what we deliver and when. Requirements are evolving, thus the delivery date might change if it is possible from business perspective.



Walker Royce, RSC2009: Uncertainty decreases throughout the project

But it does not mean that we cannot define a deadline for the delivery upfront and abide by it. We always agree on a **high level roadmap** showing how we plan to proceed. But it is not 100% commitment. Rest goes then to **continuous work with product backlog**. We know what the business objectives of the project are and we always balance ways to fulfill them. Some ways are more complex than the others. Like when building an invoicing module to the customer system, we can do fully automated solution integrated to other 3rd party's products, or we can deliver just some semi-automated solution that satisfies the basic needs and update it in next version of the product. This way we can always balance scope and complexity of the requirements in order **to fulfill business objectives and at the same time stick to the deadlines**. Sometimes the simple solutions are the best ones.

This approach should be also supported by an appropriate business model which motivates both customer and supplier to deliver on a certain date.

5. Risks

- "You are trying to transfer the risks to us by that open scope model"
- "It is too risky?"

Truth is that there is always **risk on both sides**, and no contract can change that. Fixed-all contracts push the risk to the supplier's side – when the solution is not delivered as expected, supplier is punished. But actually the customer is punished as well because he does not get the software he needs for his business competitiveness. **Lose-lose situation**.

As mentioned earlier – suppliers often try to cover this risk via buffers and financial pillows which make a solution **more expensive** and **prolong the delivery** time.

Agile approach transparency enables to handle these risks openly and tackle them together. If the risk is known and it is continuously mitigated, then its occurrence probability and impact is minimal. That is in a long term easier and cheaper solution for all involved parties.

6. Tender

In my opinion it is the most difficult situation to handle. For a couple of reasons:

- Customer already invested time in creating a very detailed requirements specifications, hence they do not want to open discussions again – *"It is all written in these documents already"*
- There are other companies participating in the tender claiming *"we can do that"* despite the fact that they can't for the proposed price, time and quality (or they do not know that they can't)

- Tender approach might be enforced by law, especially in public sector

All my articles about Agile/Lean contracts are mostly about one simple thing – we **should not commit to something we do not know** because then everyone loses if the situation gets more complicated than we expect (which usually does as there are always some **unknown unknowns**, black swans). And before we actually code some proof of concepts and tackle business and technical obstacles, we must admit that we have only limited clue about the future. This is why we should not promise something that we do not know we can deliver. Trust is hard to build but very easy to lose. So how to reduce negative impact of a tender approach?

- Can we convince the customer to use a different supplier choosing method than the tender approach? I mean, if it is not enforced-by-law?
- Can we do something in order to not commit blindly in this tender? Some proof of concept, prototype for the tender already? Can we get in touch with customer before the decision date and involve them into prototype creation? To learn more before it is too late? Working prototype might be actually a nice advantage against tender competitors. And proof of supplier competences in the area?
- Can we do a semi-commitment? To fulfill tender requirements, but not to fix everything? Like agreeing on fulfilling the business objectives but at the same time agreeing on needed support from Customer, their involvement and future easy change management?
- Tenders are not won by price only. There are many indicators that contribute to the decision about which supplier should be chosen. We can use our Agile/Lean approach as an advantage against competitors in the tender.
- The tender approach must be applied only when a certain threshold is reached. Can we agree with the customer about breaking down the functionality into smaller and cheaper chunks to avoid it?

Recently I had a discussion with a couple of customer people from one certain public sector, and they were really **sad about the law and public tenders**. They have great experience with the Agile way of working, and the current supplier (after encountering many years of failure with the traditional approach) – but the **public tender approach goes against it**. They try to avoid tendering by prolonging the current agreement they already have with the supplier they are happy with.

I understand a tender is here to save costs and prevent illegal partnerships, but on the other hand it generates huge inefficiency in areas like custom software development. Hopefully the legislative will come up with an alternate option to tender someday.

Tomáš Tureček PhD, an Lean/Agile coach in Tieto Corporation, has been working in IT industry for more than 15 years in various roles from development to management. He has more than 6 years of experience from coaching and mentoring deliveries in distributed environments. Tomáš contributes to IT communities, blogs here and speaks at prestigious international conferences.

ALM Mag - Vol. 1, Issue 5 2013

ALM Mag.com

Empowering Rapid Delivery of Quality Software

TFS Migrations
2008 & 2010 to 2012

Audio Podcasts
Technology & Process with the Experts

ALM Consulting Update

Industry Case Studies

Industry Surveys

Interviews with Industry Leaders

Best NEW Features

Event Wrap Up & Review

More
BONUS
Video

***TFS - Beyond Source Control**

Index Page