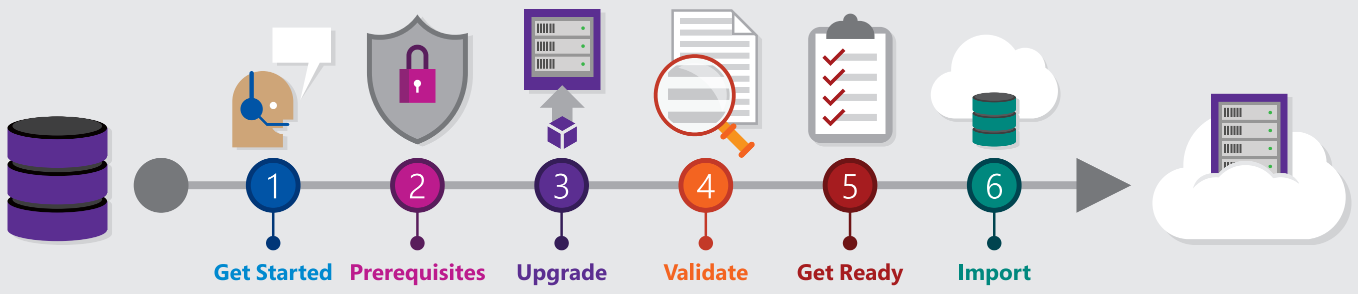
Microsoft

Team Foundation Server to Visual Studio Team Services

Migration Guide



Last updated: April 25, 2017

Download the latest version of this guide.

https://aka.ms/TFSImportData

Published: November 2016

Last updated: April 25, 2017

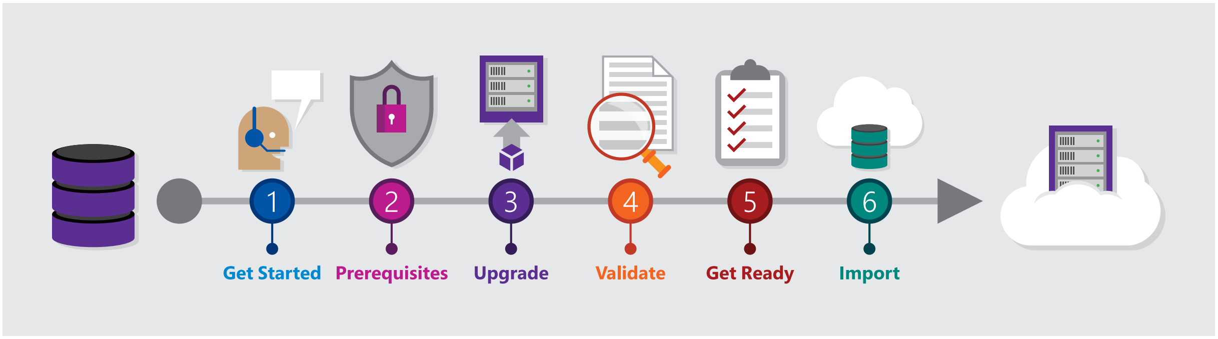
Change history

Date	Description of changes
November 2016	Initial version of the TFS to Visual Studio Team Services Migration Guide
April 2017	Removed sections that are no longer needed, minor text tweaks to make common questions on scenarios more clear, added additional technical documentation links, and corrected some broken links.

Introduction

Ever since Visual Studio Team Services was released to provide a hosted SaaS service for development teams, Team Foundation Server customers have been asking Microsoft to be able to import their TFS databases to take advantage of all the great capabilities of Visual Studio Team Services. We are happy to say that the Preview of the TFS Database Import Service for Visual Studio Team Services is now available.

This Migration Guide will walk through the different steps along the way. We have organized the guide into six phases of the migration timeline. The goal of the migration is that your team will be working in Team Foundation Server and then you will take it offline and import your database into Team Services. Your team will then be able to connect to Team Services and keep working like usual with all of the data, permissions, and customizations from your Team Foundation Server database.



This guide is not meant to replace the technical documentation for the Database Import Service but is more of a way for you to easily gather the tasks you will need to perform and the resources you will need to be successful.

Frequent updates to this guide and materials

During the Preview, we are going to be learning from customers starting their migration projects. As we both improve the Database Import Service and learn more from customer feedback, we will be updating this Migration Guide and the accompanying materials very frequently. Be sure to check back for future updates of this guide at https://aka.ms/TFSImportData.

Download the latest version of this guide.

https://aka.ms/TFSImportData

How to give feedback for this guide

If you have any suggestions for how we can make this guide better, please e-mail us at VSTSMigrationGuide@microsoft.com.

Introduction

How to find a DevOps Partner to help

We highly recommend finding a trained Microsoft Partner with the DevOps Competency, Microsoft Consulting Services, or Microsoft Premier Support to help you with your migration project to Visual Studio Team Services. The team at Microsoft has been working very closely to train the DevOps Partner community so that they can understand the different parts of migrating to Visual Studio Team Services.

Some Microsoft customers may even have help or funding available as part of their Premier Support Agreement, Enterprise Agreement, or other programs available to assist with getting help with migrating to Visual Studio Team Services and adopting Microsoft Azure. You can contact your Developer Solution Sales Specialist or Microsoft Reseller to find out more information and if you qualify.

Your Developer Solutions Sales Specialist or Microsoft Reseller can help make recommendations for getting in touch with a trained Partner to help you with your migration to Visual Studio Team Services. You can also find a list of trained DevOps parters available at https://aka.ms/FindDevOpsPartner.

☐ **Task:** Find a DevOps Partner

Find a DevOps partner.

https://aka.ms/FindDevOpsPartner

DevOps Partner Name:	
Partner Contact Info:	

Introduction

How to find your Developer Solution Sales Specialist

You may have questions about the different options available for getting access to the developer tools and services of the Microsoft Cloud including Visual Studio Team Services. Your Developer Solutions Sales Specialist or Microsoft Reseller is best equipped to give your$team the best guidance for different types of help available to you.

Reach out to your Microsoft Reseller if you have one, or if you need help with finding your Developer Solution Sales Specialist, you can reach out to us at FindYourDevSalesRep@microsoft.com.

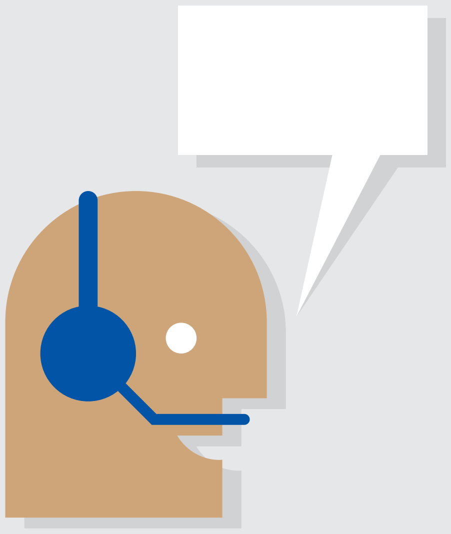
My Developer Solution Sales Specialist or Microsoft Reseller:	
Contact Info:	

How to ask questions about the migration

During your migration, you may have questions that come up. Our first recommendation is to work with a trained DevOps Partner or Microsoft Consulting Services. If you are not able to get an answer from them, you can e-mail us at VSTSDataImport@microsoft.com.

1

Get Started



Microsoft

Get Started 1

☑ Task summary

- ☐ **Choose datacenter:** Choose the datacenter for your Visual Studio Team Services account.
- ☐ **Download TFS Migrator tool:** Download the TFS Migrator tool from https://aka.ms/DownloadTFSMigrator.
- ☐ **Reserve Team Services account(s):** Reserve Team Services account(s) for each of the desired final names.
- ☐ **Fill out preview questionnaire:** Fill out preview questionnaire and request import invitation codes.

In this first phase, we are going to help you start your Visual Studio Team Services migration project and help you understand why you would want to migrate from Team Foundation Server to Visual Studio Team Services.

1 Get Started

Why migrate to Visual Studio Team Services?

If you are a Team Foundation Server customer, you have already understood how valuable it is to bring your development team together in one place with traceability from each aspect of the development lifecycle. Now that Visual Studio Team Services has been launched, many customers who have migrated from Team Foundation Server have said they did so because:

Upgraded every three weeks

Since Team Services is cloud-based, Microsoft automatically upgrades your account with the latest features as they are released. You can stay up to date on new releases at https://aka.ms/VSTSFeaturesTimeline.

VSTS Features Timeline.

https://aka.ms/VSTSFeaturesTimeline

Simplified administration

The Microsoft team will monitor your Team Services account around the clock to make sure it is available for your team. You no longer need to worry about managing the core TFS infrastructure any longer.

Accessible from anywhere

Your team members will have the flexibility they need to securely access your account from work, home, or their mobile devices. If you have remote development teams, you will have a great solution to collaborate together from anywhere.

Cloud-first innovation

Microsoft releases features to Team Services ahead of making it available in updates from Team Foundation Server. You'll be able to make your team more productive much sooner.

Included with Visual Studio subscriptions

Visual Studio (formerly MSDN) subscribers have Visual Studio Team Services already included as one of their subscription benefits.

Power your Cloud Modernization initiatives

Adopting Team Services will help your company with modernizing and driving agility and DevOps practices by making it easier to deploy your apps to the cloud and increase delivery of new business value.

Leverage developer services in the Microsoft Cloud

Using Team Services allows you to also take advantage of the many other developer services in the Microsoft Cloud like Azure, on-demand build & deployment servers, the Load Testing Service, Application Insights, HockeyApp, Xamarin Test Cloud, and many others.

Get Started 1

Ready for the Enterprise

Visual Studio Team Services is ready for teams of any size including development teams in large enterprises. There are many enterprise customers who have adopted Visual Studio Team Services and empowering their development teams to collaborate more effectively. Microsoft is also migrating to Visual Studio Team Services as its single engineering system to build its commercial, internal products, and cloud-based services.

Scalable to any team

Supports teams of any size: from tens to thousands. Backed by a 99.9% SLA and monitored by our 24x7 operations teams.

Secure by design

Core Azure services provide a secure foundation. Multi-layered security and governance technologies, operational practices, and compliance policies keep your data locked down.

Compliant

Team Services is built on Azure and has the compliance certifications many enterprises need including ISO 27001 and SOC 1 and SOC 2 compliance which demonstrate our commitment. More information is available in Phase 2 of this guide.

Azure Active Directory integration

Integration with Azure AD makes it easy to manage entire organizations. Use a common identity to access both cloud and on-premises resources. Establish and enforce password lifetime and complexity controls. Enable additional security features like multi-factor authentication.

Choice of data location

Teams can choose to host their data in different locations around the world including Europe, Australia, Brazil, India, and the United States.

1 Get Started

Fundamental differences between TFS and Team Services

Authentication

With TFS, you typically connect to a server on your on-premises network and authenticate with Windows Authentication and Active Directory. With Team Services, you'll authenticate with Azure Active Directory account credentials. To provide additional security, you can also require multi-factor authentication, IP address restrictions, conditional access, and more. In Phase 2 of this guide, you will go through the steps of setting up Azure Active Directory if your company has not implemented Azure Active Directory.

Reporting

Both TFS and Team Services have a variety of tools to give your teams insight into the progress as well as the quality of your software projects. These include:

- Dashboards and lightweight charts
- Excel reports, SQL Server Reporting Services reports, and SharePoint dashboards are available only in Team Foundation Server and not in Team Services. These powerful options have been more complicated to use.
- A Power BI connector is available only in Team Services which provides a nice combination of simplicity and power.
- REST APIs are also available for getting live data from Team Services programmatically

Note: Process Customization is now possible in Visual Studio Team Services. If your team projects in Team Foundation Server includes process template customizations, you will validate them to make sure existing customizations are supported during Phase 4 of this migration guide. Once validated, the Database Import Service will import your database including your process customizations.

Relationship between TFS databases and Visual Studio Team Services accounts

Before diving too deeply into planning your migration, it's important to understand at a high-level how the database import process functions. Imports operate on two main concepts:

Team Project Collection

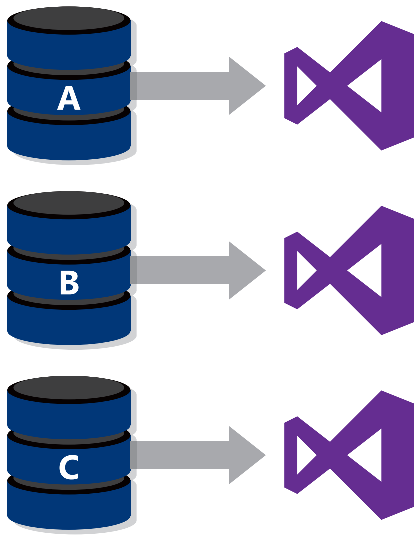
Collections in TFS are a physical container for team projects and their artifacts. Each collection equates to a single SQL database and are the source of import for migrations to Team Services.

Team Services account

Accounts are the management unit in the cloud-hosted service. Logically they map 1:1 to the concept of a team project collection in TFS. Therefore, accounts are the destination of imports for migrations to Team Services. Team Services accounts are represented as *https://contoso.visualstudio.com* where *contoso* represents the name of the Team Services account.

Note: In the future, Team Services will be introducing the concept of "organizations" that will allow your company to group multiple Team Services accounts.

Each time you import a team project collection SQL database, the Database Import Service will create a brand new Visual Studio Team Services account with a name that you provide. This means that you cannot import a collection database into an existing Team Services account or consolidate multiple collection databases into a single Team Services accounts. It is a one-to-one mapping between team project collections and Team Services accounts.



1 Get Started

Team project collection mapping worksheet

Collection name:		→	VSTS account name:		.visualstudio.com
Collection name:		→	VSTS account name:		.visualstudio.com
Collection name:		→	VSTS account name:		.visualstudio.com
Collection name:		→	VSTS account name:		.visualstudio.com
Collection name:		→	VSTS account name:		.visualstudio.com

Datacenter location

You have options for the location of your Team Services account data. The following datacenter locations are available:

United States	Europe	Australia	India	Brazil

☐ **Task:** Choose the datacenter for your Visual Studio Team Services account.

Purchases needed for Visual Studio Team Services

Another question that typically comes up is what type of licensing will a company need to adopt Visual Studio Team Services? The good news is that you are likely to have all of the licenses you already need. We have created a simplified worksheet below that you can walk through that should cover most cases. If you have any specific questions about your situation, be sure to reach out to your Developer Solution Sales Specialist or Microsoft Reseller.

User licenses worksheet

1	Number of team members	
2	Number of stakeholders	
3	Subtract Line (2) from Line (1)*	
4	Number of Visual Studio Subscribers** (formerly known as MSDN Subscribers)	
5	Subtract Line (4) from Line (3)	
6	Subtract 5 from Line (5)***	

* *Stakeholders are free.*

** *Visual Studio Subscribers have Team Services included as a benefit of the subscription.*

*** *Each Team Services account gets five free users.*

You have no charge for an unlimited number of stakeholders to be able to access your Team Services accounts without needing a user license. You also do not have a charge for any Visual Studio (formerly known as MSDN) subscribers since Team Services is included as a benefit of the subscription. Each Team Services account includes no charge for access to the core features of Team Services for the first five users.

This leaves you with the total number of Team Services user licenses that you will need to purchase to cover your team also represented by Line 6 in the worksheet above.

You will ultimately purchase any needed Team Services user licenses through the Visual Studio Marketplace or the Azure portal. We will discuss this more in Phase 5 of this guide.

You can find out more about pricing for Visual Studio Team Services at https://aka.ms/VSTSPricing and leveraging the Azure Pricing Calculator. There are additional services that you could take advantage of like hosted load testing services, Test Manager extensions, and more that would have additional costs.

Resource: Azure Pricing Calculator for Visual Studio Team Services: https://aka.ms/AzurePricingCalculatorVSTS

If you have any questions about your specific situation, you can reach out to your DevOps Partner, Microsoft Reseller, or your Microsoft Developer Solutions Sales Specialist.

1 Get Started

Download TFS Migrator tool

The bulk of the work throughout the migration to Team Services is handled by a new tool available from Microsoft called TFS Migrator. TFS Migrator will be used throughout this guide with the following high-level steps:

- Validating a TFS team project collection
- Prepare and generate the files used to customize the import
- Queueing an import of a TFS database with the Database Import Service

Download the TFS Migration Tool

https://aka.ms/DownloadTFSMigrator

You will want to download the latest version of the TFS Migrator tool and then you will run it from a Windows PC. The user who runs this tool must have:

1. The `TFSEXECROLE` role in SQL Server, and
2. Permissions to connect to both the TFS configuration and collection databases

☐ **Task:** Download the TFS Migrator tool from https://aka.ms/DownloadTFSMigrator.

Tip: The TFS Migrator tool will be continually updated based on feedback and our learnings from importing many customer TFS databases. Be sure to download the latest version of the TFS Migrator tool each week to keep up to date.

Reserve your Visual Studio Team Services account name(s)

Since the migration project make take some time to complete, you may want to "reserve" the name of your Team Services account so that the name can be available for your final import. For example, if you are from Contoso company and want a Team Services account that matches your company name like *https://contoso.visualstudio.com*, you can create an account with that name now.

However, we mentioned above that you can only import into a brand-new Team Services account. That is okay, because you when you are ready to start the final import, you could import into a Team Services account named *https://contoso-temporary.visualstudio.com* and then rename it to the desired name of *https://contoso.visualstudio.com* after deleting the originally reserved account or changing its name to something else.

Tip: If the desired name is already taken, we can help facilitate passing your contact information to the current owner of the account. They can then choose to reach out to you if they want to transfer ownership or rename their existing account so that you can create a new account with the desired name. You can reach out to us at VSTSDataImport@microsoft.com if you would like this type of help.

☐ **Task:** Reserve Team Services account(s) for each of the desired final names.

Request invitation codes for importing

During the Preview of the Team Foundation Server Database Import Service for Visual Studio Team Services, we will be reviewing which companies want to leverage the import service preview. When you are ready to queue an import using the TFS Migrator tool, you will need an invitation code that is provided to approved companies.

You can get two invitation codes by filling out a questionnaire available at https://aka.ms/vstsdataimportpreviewform. We will review your responses and reply back with two invitation codes. Why two? Many companies typically will go through a “dry run” import first to learn both the length of time required to import as well as perform some “user acceptance testing” of the imported Team Services account. They will then use the second invitation code for the final import.

If you happen to need additional invitation codes, you can reach out and request additional invitation codes at VSTSDataImport@microsoft.com. For additional dry run codes, be sure to include a previous dry run code in your mail.

☐ **Task:** Fill out preview questionnaire and request import invitation codes.

Preview questionnaire and request import invitation.

https://aka.ms/vstsdataimportpreviewform

2

Cloud Prerequisites

Microsoft

Cloud Prerequisites 2

☑ Task summary

- ☐ **Implement Azure Active Directory:** Make sure your team has a working Azure Active Directory tenant by implementing Azure Active Directory to synchronize with your on-premises Active Directory environment.

In the second phase of your migration to Visual Studio Team Services, we want to help you focus on some of the prerequisites for migrating their data to the cloud that many organizations have faced. Some of these may pertain to you and you will find the resources in the section helpful to your organization. There are other organizations who already have each of these prerequisites in place and will be able to bypass the second phase completely.

2 Cloud Prerequisites

Compliance

Visual Studio Team Services is built on all the great reliability, scalability, and standards compliance that Microsoft Azure has come to be known for. In addition to the bedrock foundation that Azure provides, Visual Studio Team Services has taken the extra steps of getting certification for individual compliance standards that customers need for their cloud-based software development services. To date, Visual Studio Team Services has the following compliance certifications:

1. ISO 27001:2013
2. SOC 1 Type 2
3. SOC 2 Type 2
4. HIPAA BAA (Business Associate Agreement)
5. EU Model Clauses

Information Security Management System
ISO
27001
Certified
AICPA Service Organization Control Reports
AICPA
SOC 2
Formerly SAS 70 Reports

The SOC audit for Team Services covers controls for data security, availability, processing integrity, andjconfidentiality.

Some of our customers who have migrated from Team Foundation Server to Visual Studio Team Services have needed to go through internal security reviews before adopting Visual Studio Team Services. We have found that the following resources were important in helping internal security teams feel comfortable with their company adopting Visual Studio Team Services.

Our internal data protection and security whitepaper

Microsoft strives for transparency about how we protect your data through multi-layered security and governance technologies, operational practices, and compliance policies. The team has documented its data protection and security practices in a whitepaper. You can learn more by reading that whitepaper at https://aka.ms/VSTSSecurity.

Data protection and security compliance policies whitepaper.

https://aka.ms/VSTSSecurity

Compliance audit report requests

The compliance audit reports are available upon request for companies who have Non-Disclosure Agreements in place with Microsoft. These audit reports were performed by our auditor, Deloitte. If you would like a copy of these reports, you can send us an e-mail at VSTSDataImport@microsoft.com or you can reach out to your Developer Solution Specialist.

Cloud Prerequisites

Azure Active Directory

The main task for Phase 2 is to make sure your team has a working Azure Active Directory tenant that will be used for authenticating your team members in your Team Services account.

☐ **Task:** Implement Azure Active Directory to synchronize with your on-premises Active Directory environment.

User authentication in Team Foundation Server is handled on-premises by using Active Directory. With Visual Studio Team Services, users are authenticated through an Azure Active Directory tenant which works very similarly to Active Directory on-premises. In Phase 5, you will be verifying an identity map file that will map your on-premises Active Directory accounts to matching Azure Active Directory accounts. This will allow each of your team members to see their individual history, preserve security permissions, and make sure they have access to all of their personal settings including favorites, personal queries, etc.

Microsoft Accounts (or MSAs) are available to use for authentication with Team Services but not available for mapping when you are importing a Team Foundation Server database using the Database Import Service.

Many companies who leverage services from the Microsoft Cloud including Office 365 and Azure, already have an Azure Active Directory tenant setup that is synchronizing user accounts and groups from Active Directory on-premises. Our recommendation is to not setup a separate Azure AD tenant for your Team Services implementation. You will want to use the same Azure Active Directory tenant as other Microsoft Cloud services at your company.

2 Cloud Prerequisites

If your company already has Azure Active Directory available, then you can skip this step in this guide and move forward.

Synchronizing identities and groups with Azure AD Connect

By synchronizing your on-premises Active Directory with Azure Active Directory, your team members will be able to use the same credentials to authenticate and your Team Services administrators will be able to leverage your Active Directory groups for setting permissions within your Team Services account.

To setup the synchronization, you will want to use the Azure AD Connect technology. You will likely want to work with your IT department, your DevOps Partner, Microsoft Premier Support, or Microsoft Consulting Services to help set up Azure AD Connect with your on-premises environment.

The documentation for setting up Azure AD Connect is available at https://aka.ms/AzureADConnect.

Note: DirSync was a predecessor technology to Azure AD Connect. You will want to upgrade to Azure AD Connect if you are using DirSync.

To read more about how Team Services can be set up to use Azure Active Directory, you can visit: https://aka.ms/AADforVSTS. Since you will be importing your TFS database, you will not be following the steps exactly in that article but it is good reference information for how it works. The TFS Database Import service will set up the link to your Azure Active Directory tenant when your Team Services account is created as part of the beginning of the Database Import service process.

Cloud Prerequisites 2

Additional security for Cloud authentication

Once you have Azure Active Directory setup, we have seen many customers take additional steps that are available natively with Azure Active Directory to provide additional security measures for access to development team data as well as other data in Microsoft Cloud services like Office 365 and Azure. The next sections cover optional additional steps you can take to further secure your Team Services account.

Multi-Factor Authentication

One of the main additional security mechanisms that our customers have added is taking advantage of Multi-Factor Authentication (MFA) requirements as part of getting access to the data stored in a Team Services accounts. Two-step verification is a method of authentication that requires more than one verification method and adds a critical second layer of security to user sign-ins and transactions. It works by requiring any two or more of the following verification methods:

- Something you know (typically a password)
- Something you have (a trusted device that is not easily duplicated, like a phone)
- Something you are (biometrics)

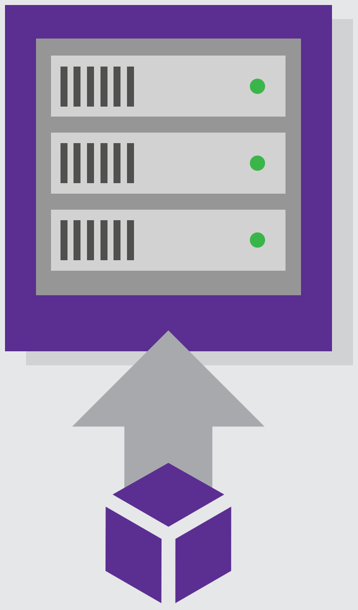
You can learn more about setting up Multi-Factor Authentication requirements with Azure Active Directory here: https://aka.ms/AzureADMFA

Conditional Access

The other common security practice we see with teams adopting Team Services is to set conditional access rules in Azure Active Directory that provide for additional security mechanisms based on which applications they are signing into and from what location they are signing-in from. For example, you may want to specify that accessing Team Services always requires MFA or that MFA is only required if your team member is accessing Team Services from outside of the office (i.e., when they are at home, from their phone, or traveling).

Conditional Access capabilities allow for powerful combinations of security policies based on your organization’s needs. You can find more information about setting up Azure Conditional Access here: https://aka.ms/AzureConditionalAccess

3 Upgrade TFS



Microsoft

Upgrade TFS 3

☑ Task summary

- ☐ **Upgrade your Team Foundation Server:** Upgrade your Team Foundation Server to one of the supported versions.
- ☐ **Run “Configuration Features”:** Run the “Configure Features” wizard on every team project in each of your team project collections.

One of the major prerequisites for migrating your Team Foundation Server database is to get your database schema version as close as possible to what is currently deployed in Visual Studio Team Services. In Phase 3 of your migration project, you will work on upgrading your Team Foundation Server to one of the supported versions for the Database Import Service in Visual Studio Team Services.

3 Upgrade TFS

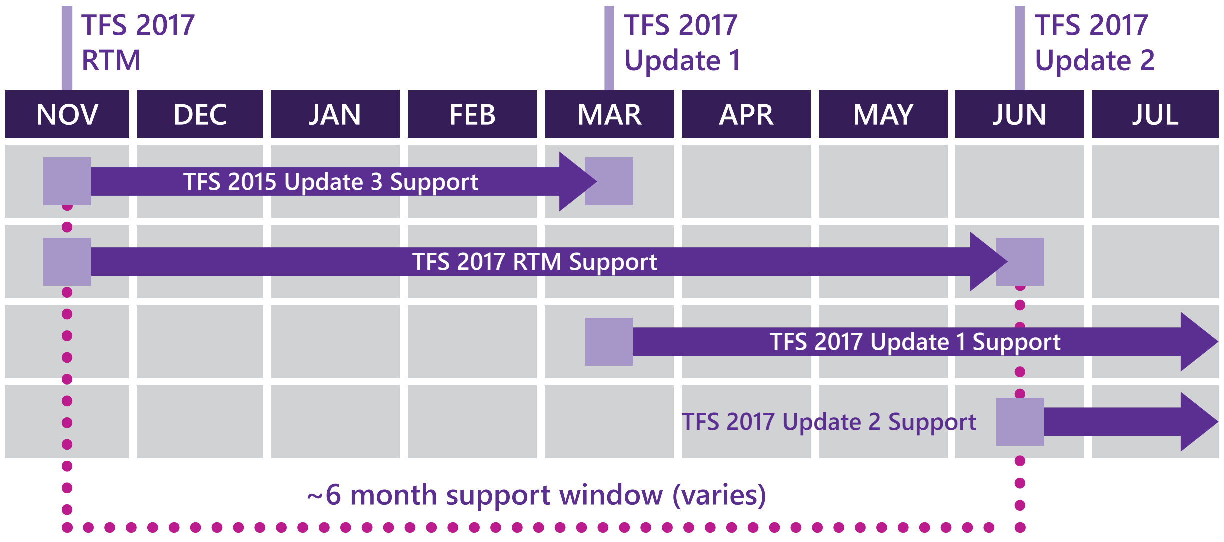
Timeline of support for TFS versions

It is important to note that the TFS Database Import Service for Visual Studio Team Services does not support all versions of TFS databases. At any given time, the Database Import service will support the current version of TFS and the previous version. Updates are included in the timeline for supported versions.

We have included a visual representation of how long individual releases will be supported as soon as new versions come out. We do not currently have release dates for future TFS updates so we have added sample release dates as examples to help you understand the impact on the supported database versions timeline.

See which versions of TFS are currently supported at https://aka.ms/VSTSImportSupportedVersions

Sample release schedule


TFS 2017 RTM
TFS 2017 Update 1
TFS 2017 Update 2
NOV
DEC
JAN
FEB
MAR
APR
MAY
JUN
JUL
TFS 2015 Update 3 Support
TFS 2017 RTM Support
TFS 2017 Update 1 Support
TFS 2017 Update 2 Support
~6 month support window (varies)

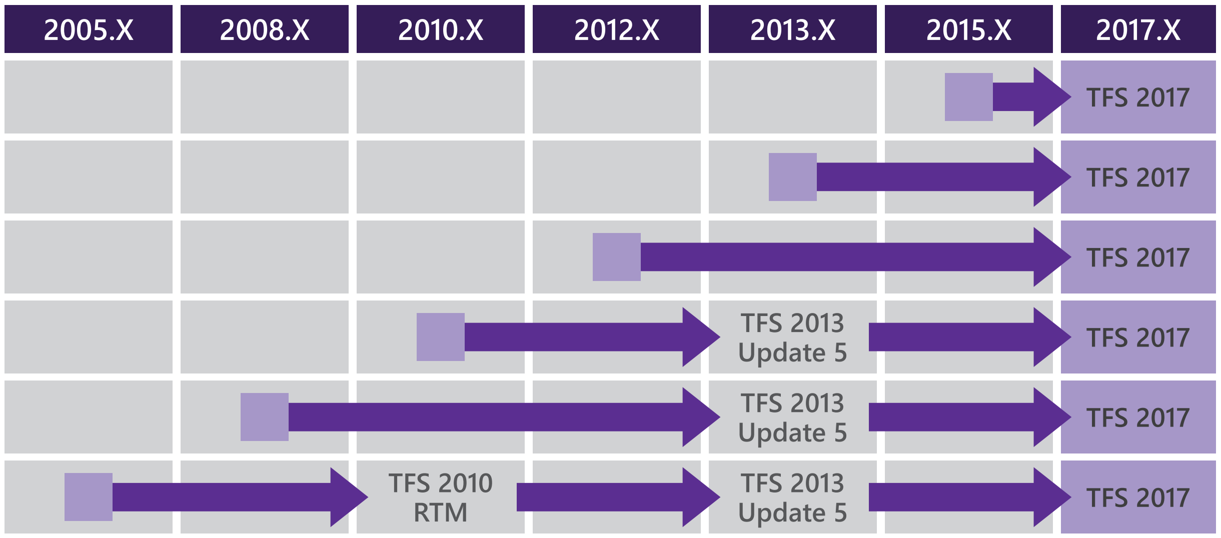
Sample release schedule—dates are examples and not actual.

Note: It is important to look ahead in the entire migration process. If you need to upgrade, we suggest that you upgrade to the latest version of TFS at any given time since you may have a longer migration project cycle than other customers.

Tip: It's required that you complete at least one "dry run" import for your database. You should include padding in your migration timeline for at least one "dry run" database import attempt to test out the dry run imported Team Services account. We will cover that in Phase 5 and Phase 6 of this guide but it is good to understand this now so that you can plan accordingly for upgrading your TFS server and then migrating to Team Services before you will need to upgrade your server again.

Upgrade paths

Microsoft has been releasing Team Foundation Server for over 10 years now so many customers are on various versions. Your goal will be to get to the latest version of TFS as mentioned in the previous section. Depending on which version of TFS you currently have in production, you have a few different paths to get you to the latest version of TFS. However, TFS 2017 does not allow you to have a single-step upgrade from every version of TFS in the past so your upgrade path may include a few interim steps along the way. We have summarized each of the upgrade paths in the diagram below from which TFS version you currently have installed.



If you are using TFS 2012, TFS 2013, or TFS 2015, upgrade it directly to TFS 2017. If your TFS deployment is on an earlier version, you will need multiple steps. See the illustration above for our recommended steps.

3 Upgrade TFS

Tip: TFS Updates are self-contained as of TFS 2012. As such, there is no need to upgrade to an RTM version of TFS and then apply the update – just upgrade to the update version directly. See TFS System Requirements for Dependencies.

One thing to remember as you plan for upgrades for your Team Foundation Server environment are the underlying system requirements of the dependencies of TFS at different TFS versions. TFS has several dependencies that you will need to verify are still supported along your upgrade path:

- Operating System
- SQL Server
- SharePoint
- Project Server
- Visual Studio IDE
- Office
- Team Foundation Server Build Agent

There is a full list of system requirements for every version of TFS available for your reference at https://aka.ms/TFSSystemRequirements.

Upgrading Team Foundation Server

Now that you know what your upgrade path looks like for your Team Foundation Server environment, you can start the steps of upgrading. This is when you will likely want to work with your selected DevOps Partner, Microsoft Consulting Services, or Microsoft Premier Support to help you with planning out the upgrade portion of your Visual Studio Team Services migration project. Each of these partners are well trained in the steps necessary to upgrade your Team Foundation Server environment.

☐ **Task:** Upgrade your Team Foundation Server.

Upgrade resources

For your convenience, we are including each of the Upgrade Guides for the different Team Foundation Server upgrades you may need to perform given the upgrade paths above.

- TFS 2017 Upgrade Guide: https://aka.ms/TFS2017Upgrade
- TFS 2013 Update 5 Upgrade Guide: https://aka.ms/TFS2013Upgrade
- TFS 2010 Upgrade Guide: https://aka.ms/TFS2010Upgrade

TFS 2017 Upgrade Guide.

https://aka.ms/TFS2017Upgrade

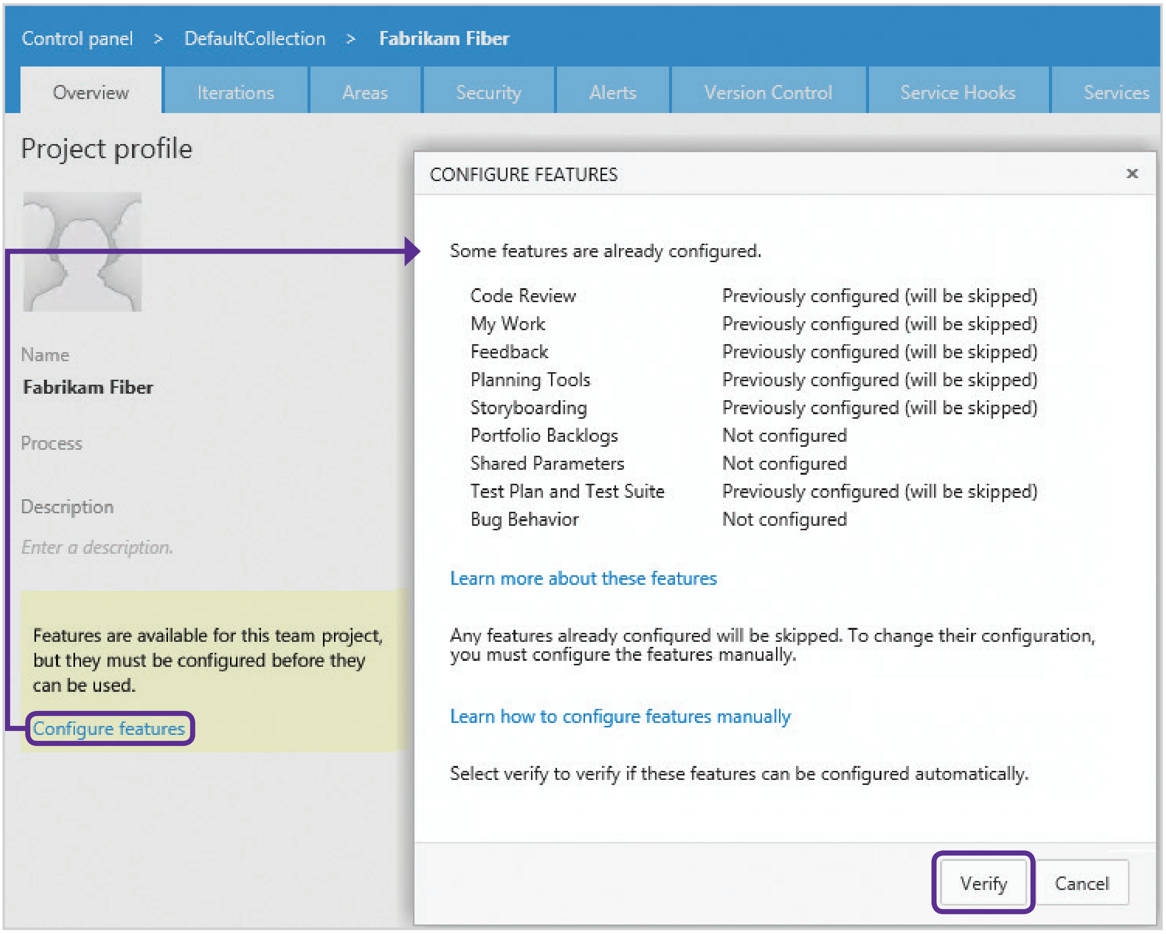
Post-upgrade steps

Now that you have upgraded your Team Foundation Server to TFS 2017, there are some additional post-upgrade tasks that we highly suggest taking before you move to the next phase of your Visual Studio Team Services migration project.

Configure Features wizard

☐ **Task:** Run the "Configure Features" wizard on every team project in each of your team project collections.

The process used by your team projects does not get upgraded along with your collection databases. Instead, you'll need to run the *Configure Features* wizard to incorporate process changes that enable new functionality like agile planning tools and code reviews. This step is an important part of migrating to Visual Studio Team Services, since it helps to ensure that the processes used in your team projects conform to the requirements of the Database Import Service. To find out more about how to use the "Configure Features" wizard, you can find the documentation article at https://aka.ms/TFSConfigureFeatures.



Features are available for this team project, but they must be configured before they can be used.

Some features are already configured.

Code Review	Previously configured (will be skipped)
My Work	Previously configured (will be skipped)
Feedback	Previously configured (will be skipped)
Planning Tools	Previously configured (will be skipped)
Storyboarding	Previously configured (will be skipped)
Portfolio Backlogs	Not configured
Shared Parameters	Not configured
Test Plan and Test Suite	Previously configured (will be skipped)
Bug Behavior	Not configured

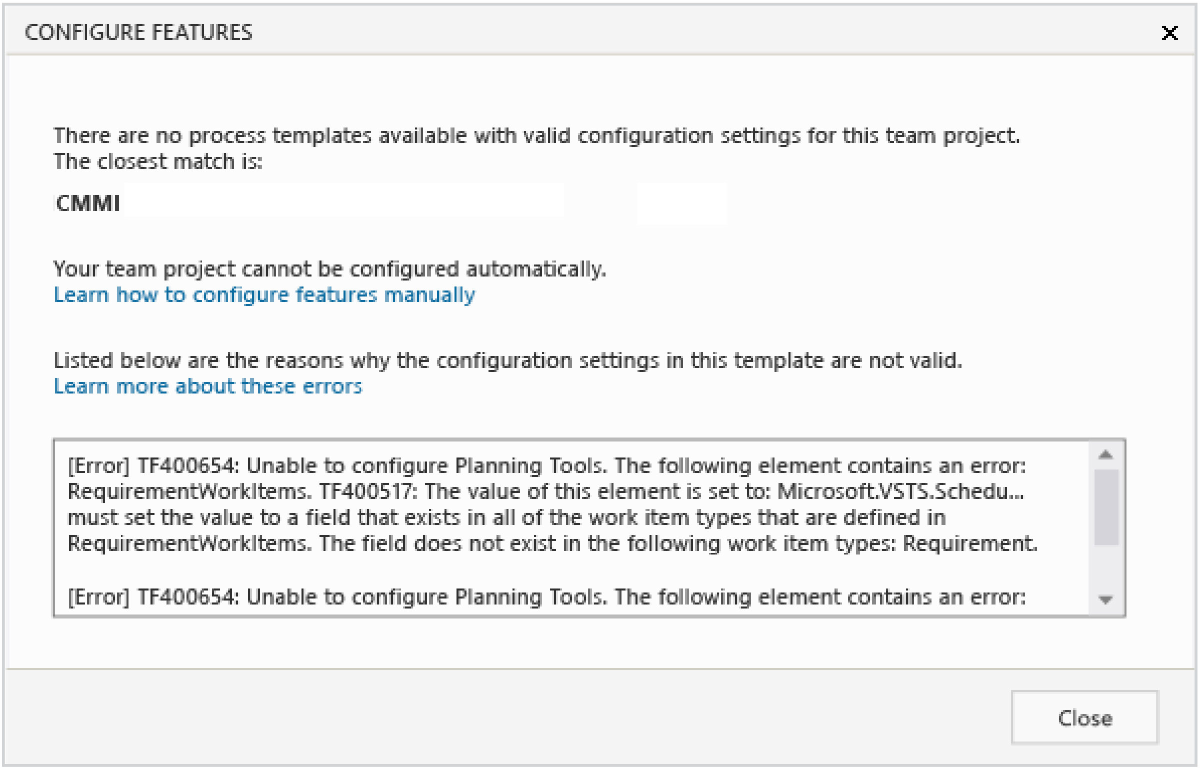
Any features already configured will be skipped. To change their configuration, you must configure the features manually.

Select verify to verify if these features can be configured automatically.

Upgrade TFS

Applying process template updates manually

If you have heavily customized your process or have used third party process templates, the “Configure Features” wizard may not be able to automatically configure features for your team projects. In these cases, you will need to configure features manually. See the section in the documentation article titled “Apply updates manually” in https://aka.ms/TFSConfigureFeatures for more information. If you find yourself in this situation, we highly recommend working with a trained DevOps Partner, Microsoft Consulting Services, or Microsoft Premier Support.



There are no process templates available with valid configuration settings for this team project.
The closest match is:

CMMI

Your team project cannot be configured automatically.
Learn how to configure features manually

Listed below are the reasons why the configuration settings in this template are not valid.
Learn more about these errors

```
[Error] TF400654: Unable to configure Planning Tools. The following element contains an error:
RequirementWorkItems. TF400517: The value of this element is set to: Microsoft.VSTS.Schedu...
must set the value to a field that exists in all of the work item types that are defined in
RequirementWorkItems. The field does not exist in the following work item types: Requirement.

[Error] TF400654: Unable to configure Planning Tools. The following element contains an error:
```

4

Validate Your TFS Server

Microsoft

4 Validate Your TFS Server

☑ Task summary

- ☐ **Run validations with TFS Migration tool:** Run the validation of each team project collection database with the TFS Migrator tool.
- ☐ **Review logs and fix errors:** Review the logs and fix any errors that were found.
- ☐ **Send us feedback:** Please send your logs to vstsdataimport@microsoft.com so we can learn from them regarding additional areas to invest in to make the TFS Migrator tool better.
- ☐ **Repeat validation checks:** Repeat the validation and error fixing process until there are no more errors remaining in the logs.

Now that you have your Team Foundation Server environment upgraded to the latest version, you will start the work of ensuring that it is ready to import. The focus of Phase 4 is using the TFS Migrator tool to run verification steps to discover any errors. This section of the guide will also help you troubleshoot some common errors that may come up as well as what to do to fix them.

If you have not downloaded the latest version of the TFS Migrator tool, refer to Phase 1 of this guide for where to download it. It is important to check each week for a new build of the TFS Migrator tool since Microsoft will be continually improving it based on feedback and learning from customer scenarios and TFS environments.

Validate Your TFS Server 4

Validating a team project collection

Since each team project collection is its own SQL database in TFS, you will run the validation process on each team project collection in your environment. Validation will examine a variety of aspects of your collection, including:

- Size of your collection database
- Collation of the SQL database
- Identities of users in the collection
- Analyze process template customizations

☐ **Task:** Run the validation of each team project collection database with the TFS Migrator tool.

You begin a validation by using the TFS Migrator tool. We recommend that you run the TFS Migrator tool from one of the application tier (AT) servers for your TFS environment. You can find out more about the specific command-line options by requesting the help text with the command below.

```
TfsMigrator validate /help
```

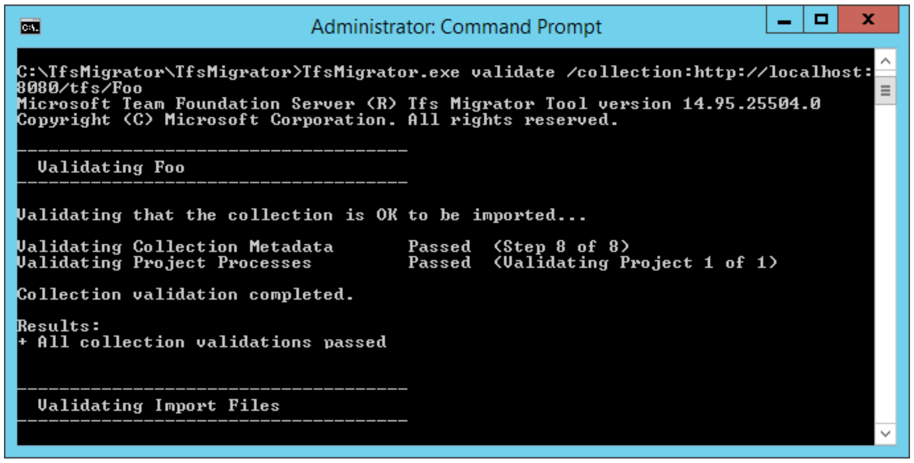
The most common way to start a validation is to specify the URL of the team project collection with the command below.

```
TfsMigrator validate /collection:http://localhost:8080/tfs/
DefaultCollection
```

There is additional technical documentation available for the validation phase available at https://aka.ms/VSTSValidateCollection.

Review validation warnings and errors

Once the TFS Migrator tool is finished, there will be a set of log files and a set of results printed to the command prompt screen.



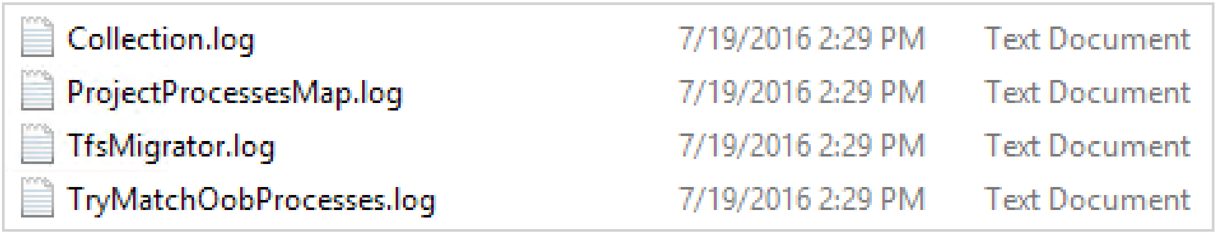
```
C:\TfsMigrator\TfsMigrator>TfsMigrator.exe validate /collection:http://localhost:
8080/tfs/Foo
Microsoft Team Foundation Server (R) Tfs Migrator Tool version 14.95.25504.0
Copyright (C) Microsoft Corporation. All rights reserved.

-------------------------------------
  Validating Foo
-------------------------------------

Validating that the collection is OK to be imported...

Validating Collection Metadata       Passed  (Step 8 of 8)
Validating Project Processes         Passed  (Validating Project 1 of 1)

Collection validation completed.

Results:
+ All collection validations passed


-------------------------------------
  Validating Import Files
-------------------------------------
```

4 Validate Your TFS Server

If there were no errors and all the validation checks have passed, then your team project collection is ready and you can move on to the next phase. If it does not say that all of the validation checks have passed, then you will need to look through the log files to find any errors and fix them.

☐ **Task:** Review the logs and fix any errors that were found.

There are a set of logs that are generated during the validation phase. The main log that you will want to focus on is the `TfsMigrator.log` file which contains the main details on the validation checks that were run. The other files exist to contain only the errors in the section of the validation checks that match their file name. The `TryMatchOobProcessMatch.log` should be ignored if you have applied any customizations to your team project's process templates.



```
Collection.log             7/19/2016 2:29 PM   Text Document
ProjectProcessesMap.log    7/19/2016 2:29 PM   Text Document
TfsMigrator.log            7/19/2016 2:29 PM   Text Document
TryMatchOobProcesses.log   7/19/2016 2:29 PM   Text Document
```

☐ **Task:** If you do hit any errors, we would ask that you send your logs to vstsdataimport@microsoft.com so that we can learn from them on additional areas where we should invest in making the TFS Migrator tool better.

There are several types of errors that could show up in the logs from the validation checks. Solutions for many of the errors are being documented in our troubleshooting guide at https://aka.ms/VSTSMigrationTroubleshooting. You should also leverage your trained DevOps Partner, Microsoft Consulting Services, or Microsoft Premier Support to help you with solutions for each of the errors you encounter.

Process template errors

The most common types of errors that we have seen have been process template errors that are either because the latest features of TFS have not been added to older team projects or there are customizations that Visual Studio Team Services does not support now. There are many customizations that Team Services does support so the validation checks only look for customizations that need to be fixed before migrating to Team Services.

A list of supported process customizations is available at https://aka.ms/SupportedProcessCustomizations

Validate Your TFS Server 4

At the end of Phase 3, you ran the "Configure Features" wizard on each of the team projects in your collections so you should not have any errors related to missing process template items from newer features of TFS.

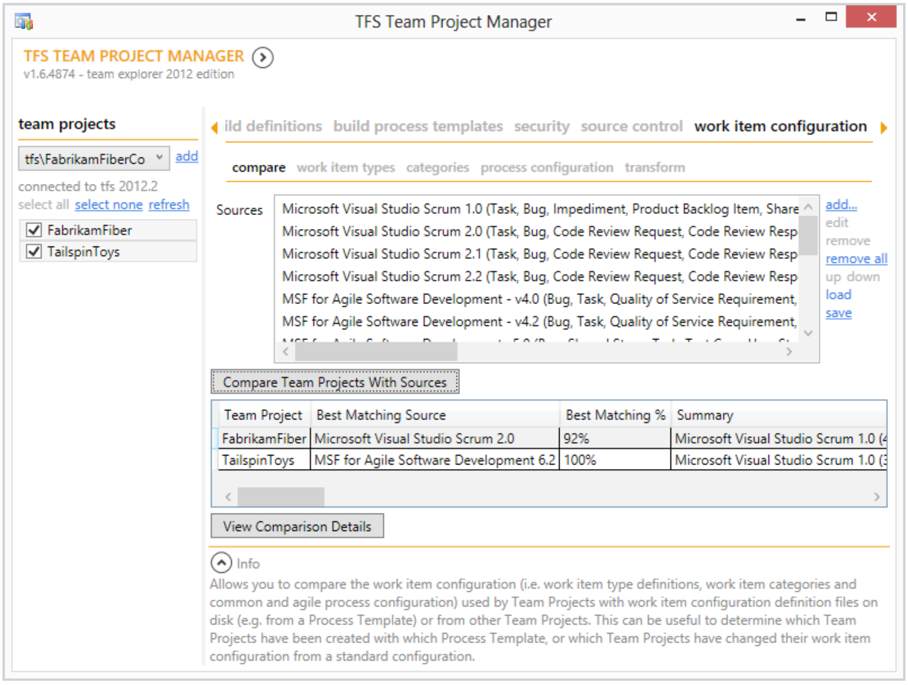
For the remaining types of process errors, you will use the `witadmin.exe` command-line tool that is included with installations of Visual Studio. There is deeper technical documentation for addressing many of the process errors that show up in the validation logs at https://aka.ms/VSTSProcessErrors.

There are a few tips for tools you can use to help you with addressing process errors in addition to `witadmin.exe`.

To help with troubleshooting process template errors, you may want to automate exporting the process templates for each of the team projects in your team project collection. There is an undocumented command for the TFS Migrator tool that will help you out. You can add this option at the end of the `validate` command to generate zip files of each of the process templates used by each of the team projects.

```
TfsMigrator validate /collection:http://localhost:8080/tfs/
DefaultCollection /SaveProcesses
```

Another tool that many TFS administrators find helpful in this scenario is the TFS Team Project Manager available on CodePlex at https://aka.ms/TeamProjectManager. One of the most useful features of this tool is the ability to compare each team project with known process templates (like the out of the box process templates). You can then look at the comparison details for the work item types and project process configuration settings to see what is different.


TFS Team Project Manager
TFS TEAM PROJECT MANAGER
v1.6.4874 - team explorer 2012 edition
team projects
tfs\FabrikamFiberCo
add
connected to tfs 2012.2
select all select none refresh
FabrikamFiber
TailspinToys
ild definitions build process templates security source control work item configuration
compare work item types categories process configuration transform
Sources
add...
edit
remove
remove all
up down
load
save
Compare Team Projects With Sources
View Comparison Details
Info

Microsoft Visual Studio Scrum 1.0 (Task, Bug, Impediment, Product Backlog Item, Share
Microsoft Visual Studio Scrum 2.0 (Task, Bug, Code Review Request, Code Review Resp
Microsoft Visual Studio Scrum 2.1 (Task, Bug, Code Review Request, Code Review Resp
Microsoft Visual Studio Scrum 2.2 (Task, Bug, Code Review Request, Code Review Resp
MSF for Agile Software Development - v4.0 (Bug, Task, Quality of Service Requirement,
MSF for Agile Software Development - v4.2 (Bug, Task, Quality of Service Requirement,

Team Project	Best Matching Source	Best Matching %	Summary
FabrikamFiber	Microsoft Visual Studio Scrum 2.0	92%	Microsoft Visual Studio Scrum 1.0 (4
TailspinToys	MSF for Agile Software Development 6.2	100%	Microsoft Visual Studio Scrum 1.0 (3

Allows you to compare the work item configuration (i.e. work item type definitions, work item categories and common and agile process configuration) used by Team Projects with work item configuration definition files on disk (e.g. from a Process Template) or from other Team Projects. This can be useful to determine which Team Projects have been created with which Process Template, or which Team Projects have changed their work item configuration from a standard configuration.

4 Validate Your TFS Server

Collection size

The TFS Database Import Service for Visual Studio Team Services can import very large databases but if your database is over 150 GB then we will have an alternate process from what is described in this guide. Please contact us at vstsdataimport@microsoft.com if you see this warning. There is also an alternate method that you will use to create a backup of your SQL database that is discussed in Phase 6.

SQL Database collation

There are currently only two collations that are supported in the preview of the TFS Database Import Service. Those two collations are:

- SQL_Latin1_General_CP1_CI_AS
- Latin1_General_CI_AS

If you have a different collation, it's still possible to import. See the following page for more details: https://aka.ms/VSTSImportCollations

Repeating the validation checks

There will be a few iterations where you will resolve some errors and then repeat running the validation checks to see if the error is no longer detected in the validation log files. You will want to repeat this process until there are no more errors and you see the success confirmation that all collection validation checks have passed.

☐ **Task:** Repeat this validation and error fixing process until there are no more errors remaining in the logs.

5 Get Ready for Import

5 Get Ready for Import

☑ Task summary

- ☐ **Assign, activate, and map Visual Studio subscriptions:** Ensure that each of the Visual Studio (formerly MSDN) subscriptions are assigned, activated, and mapped to each subscriber's Azure Active Directory account.
- ☐ **Generate import settings:** Generate import settings and related files using the `TfsMigrator prepare` command.
- ☐ **Provide the configurable settings:** Provide the configurable settings in the Import Specification file.
- ☐ **Complete and verify the Identity Map**
- ☐ **Verify and update licenses in the Identity Map**
- ☐ **Task:** Create an Azure Storage Container in the same datacenter as the final Visual Studio Team Services account.

Now that you have confirmed that your Team Foundation Server collection database is validated, your team can start to prepare for your dry run and final imports. This section of the guide is dedicated to preparing your team and generating the files that are needed by the TFS Database Import Service in Visual Studio Team Services.

Get Ready for Import 5

Subscriptions

One of the import files that will be generated is an identity map which among other things includes a licensing column. You may remember from Phase 1 of this guide that Visual Studio (formerly MSDN) subscribers include access to Visual Studio Team Services as a benefit of their subscription. Taking advantage of that benefit requires that each subscription is assigned, activated, and mapped to the Azure Active Directory account for the subscriber if the subscription is not assigned to the Azure Active Directory account from the beginning. This will likely take some time since each of your team members will potentially have action items and you will want to work with your company's designated Visual Studio Subscriptions Administrator (formerly MSDN Administrator).

☐ **Task:** Ensure that each of the Visual Studio (formerly MSDN) subscriptions are assigned, activated, and mapped to each subscriber's Azure Active Directory account.

The high-level set of steps for each subscription your team owns are:

1. The Visual Studio Subscriptions Administrator logs into the Administrator's Portal and assigns a subscription to each of the team members. The recommended approach for this step is to assign the subscription to the Azure Active Directory account of the subscriber.
2. The subscriber then goes to the subscriber portal and logs in with the same e-mail address to activate the subscription.
3. If the subscription was activated using a Microsoft Account (MSA), then the subscriber will need to link their Azure Active Directory account to their subscription so that Visual Studio Team Services will recognize the subscriber's benefit when they login to Team Services with their Azure Active Directory account.

Assign subscription

In the first step, the Subscriptions Administrator for your company will log in to the Administrator's Portal (https://aka.ms/VSSubscriptionAdminPortal) and assign each of the available subscriptions to the relevant team members.

The new portal supports assigning a subscription to an Azure Active Directory account, so we highly recommend that the administrator assigns the subscription to the Azure Active Directory account for the subscriber (i.e., johndoe@contoso.com) instead of assigning to a personal Microsoft Account (johndoe@outlook.com).

Get Ready for Import

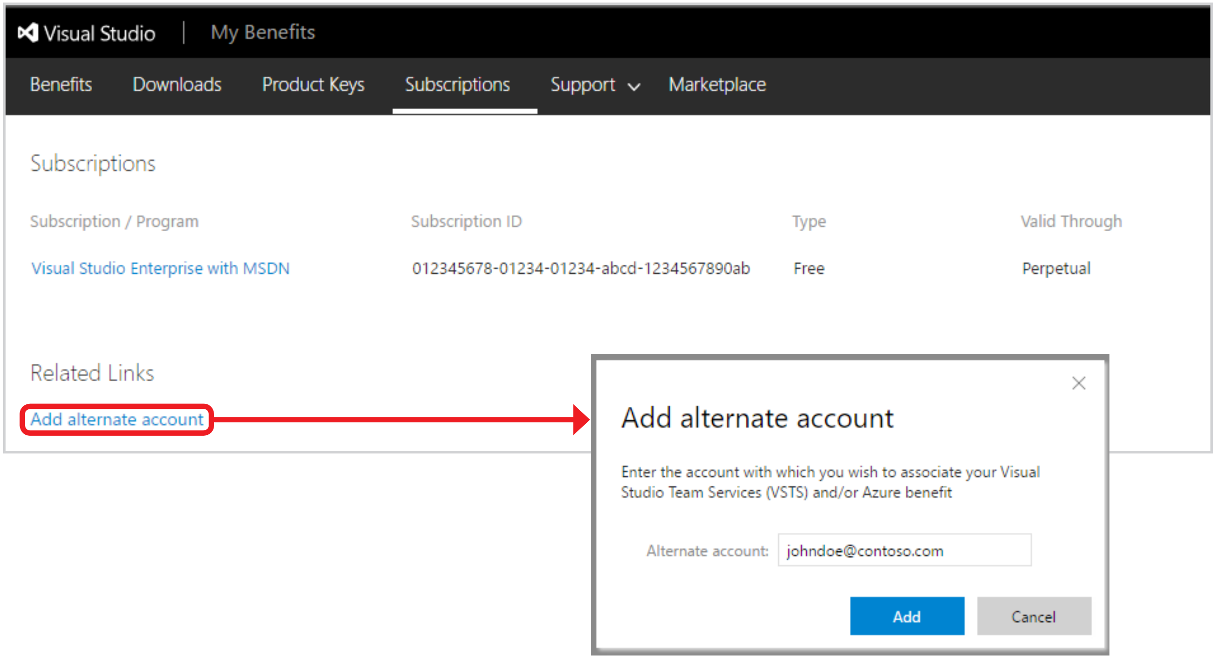
Activate subscription

Each subscriber will then login to the Visual Studio Subscriptions Portal at https://my.visualstudio.com with the account that was assigned. If the administrator assigned the subscription to the subscriber's Azure Active Directory account, then they need to sign-in with their Azure Active Directory account.

Link subscription to Azure Active Directory account

Many legacy subscribers will find that their subscription was assigned to and activated with a Microsoft Account. The last step for each subscriber to take is to link the Visual Studio Subscription to their Azure Active Directory account. There are a few different methods for doing this step which are documented at https://aka.ms/LinkVSSubscriptionToAADAccount.

Note: This is a very important step for each subscriber to complete before the final production import of your TFS database, and ideally before you generate the identity map later in this section.



Subscriptions

Subscription / Program	Subscription ID	Type	Valid Through
Visual Studio Enterprise with MSDN	012345678-01234-01234-abcd-1234567890ab	Free	Perpetual

Related Links

Add alternate account

Add alternate account

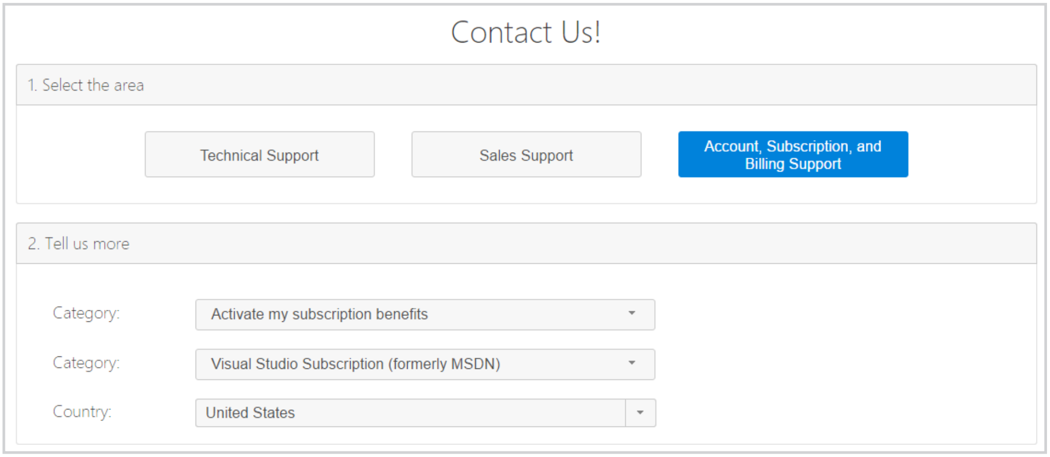
Enter the account with which you wish to associate your Visual Studio Team Services (VSTS) and/or Azure benefit

Alternate account: johndoe@contoso.com

Get Ready for Import

Help with subscriptions

If your team needs any help with activating the benefits of your subscriptions, you can reach out to the Support team at https://aka.ms/VSSubscriptionHelp. When creating a new support case, choose “Account, Subscription, and Billing Support” and then “Activate my subscription benefits” to get started.



Category:	Activate my subscription benefits
Category:	Visual Studio Subscription (formerly MSDN)
Country:	United States

Generate import files with prepare step in TfsMigrator

You are ready to generate the import specification and related files you will need to queue an import of your TFS collection database. This section will cover the different files that are produced but first you will need to run the prepare command to generate them.

```
TfsMigrator prepare /collection:http://localhost:8080/tfs/DefaultCollection
/tenantDomainName:contoso.com
```

The tenant domain name option is the name of your company’s Azure Active Directory tenant. The prepare command will contact your Azure Active Directory tenant so it will prompt you to login with a user from the tenant with permissions to read information about all of the users in the Azure Active Directory tenant. It is important to understand that the prepare command needs to have access to the Internet for this step. If your TFS server does not have access to the Internet, then you will need to run this command from a different computer.

More information about the prepare command is available at https://aka.ms/TfsMigratorPrepare.

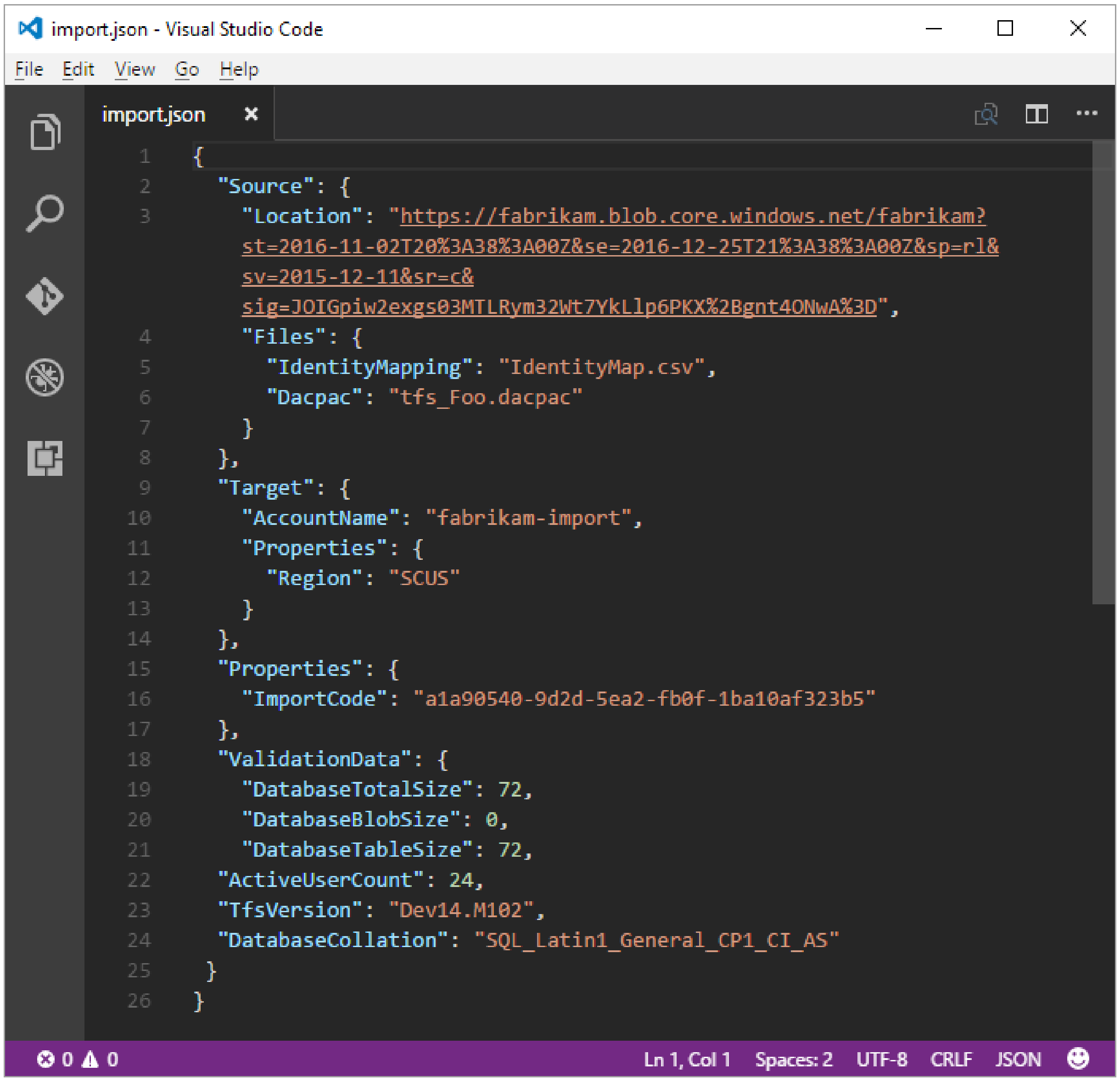
☐ **Task:** Generate import settings and related files using the `TfsMigrator prepare` command.

5 Get Ready for Import

Import specification file

The import specification file is a JSON file that will instruct the TFS Database Import service how to configure your imported Visual Studio Team Services account, specify the source file locations, and customize the import.

☐ **Task:** Provide the configurable settings in the Import Specification file.



```json
{
  "Source": {
    "Location": "https://fabrikam.blob.core.windows.net/fabrikam?
    st=2016-11-02T20%3A38%3A00Z&se=2016-12-25T21%3A38%3A00Z&sp=rl&
    sv=2015-12-11&sr=c&
    sig=JOIGpiw2exgs03MTLRym32Wt7YkLlp6PKX%2Bgnt4ONwA%3D",
    "Files": {
      "IdentityMapping": "IdentityMap.csv",
      "Dacpac": "tfs_Foo.dacpac"
    }
  },
  "Target": {
    "AccountName": "fabrikam-import",
    "Properties": {
      "Region": "SCUS"
    }
  },
  "Properties": {
    "ImportCode": "a1a90540-9d2d-5ea2-fb0f-1ba10af323b5"
  },
  "ValidationData": {
    "DatabaseTotalSize": 72,
    "DatabaseBlobSize": 0,
    "DatabaseTableSize": 72,
  "ActiveUserCount": 24,
  "TfsVersion": "Dev14.M102",
  "DatabaseCollation": "SQL_Latin1_General_CP1_CI_AS"
 }
}
```

Get Ready for Import 5

Several of the fields are auto-populated during the prepare step but some will need to be configured by you. The fields that you will need to provide are:

- **Account Name:** the name of the Visual Studio Team Services account that you want to be created for importing your data.
- **Region:** the datacenter you want the Team Services account to be created in from your choice in Phase 1 of this guide.
- **Location:** a backup of your database and import files will be uploaded to an Azure storage container. This field specifies the SAS key that will be used by the TFS Database Import Service to securely connect to and read the source files from the Azure storage container. Creating the storage container will be covered later in Phase 5 and generating a SAS key will be covered in Phase 6 before you queue a new import.
- **Dacpac:** a file that packages up your collection's SQL database.
- **Identity Mapping:** the filename of your identity map.

Beginning with the Public Preview of the TFS Database Import Service, you will also need to specify an import invitation code in the space provided. This is one of the invitation codes that you requested from the Microsoft team after filling out the Preview questionnaire in Phase 1 of this guide.

More information about the import specification file can be found at https://aka.ms/VSTSImportSpecification.

Identity Map

The identity map is one of the most important configuration files for the TFS Database Import service since it is what maps the on-premises Active Directory identities with a matching Azure Active Directory identity. This is so that each of your team members will automatically have their personal settings, security permissions, and history tied to their Azure Active Directory user account which makes for a positive experience working in Visual Studio Team Services on the first day.

☐ **Task:** Complete and verify the Identity Map.

5 Get Ready for Import

Historical vs. active identities

When importing an identity, the Import service will decide whether the identity will become *active* or *historical*.

Note: It is important to note that once an identity is imported as a historical identity, there is no way to transition that identity to become active again in the future.

Active identities

Identities that will be active users in Visual Studio Team Services after imported. Active identities will have a license and show up as a user in the account after migration. Active identities are included in the mapping file. If you want an identity to be active post-migration, make sure to include a correct mapping in the identity mapping file.

Historical identities

Identities that are not specified in the identity mapping file or the identity mapping line is not completely filled out. Historical identities do not have access to the Team Services account post-migration, do not have licenses, and do not show up as a user in the account. They will be included in the history of data like version control, work items, builds, and other places where they have contributed in the past. Historical identities are recommended for users that are no longer at the company or will not ever be needing access to the Team Services account again. Historical identities **cannot** be converted to an active identity in the future.

Understanding the identity map

The identity map is a Comma Separated Value (CSV) file which details how your identities will be imported to Team Services. For an explanation of the collections see the following page: https://www.visualstudio.com/en-us/articles/migration-import#identity-map

Specify licenses

☐ Task: Verify and update licenses in the identity map

One of the main steps that you will want to do is verify that all the licenses are specified correctly and override any that should be overridden with a different value. Details on how to override the licensing values in the identity mapping file can be found at https://aka.ms/VSTSIdentityMapOverride

This is a good chance to tell which of your Visual Studio (formerly known as MSDN) subscribers have not linked their subscriptions to their Azure Active Directory account. More information about this linking process can be found at the beginning of Phase 5 in this guide.

The good news is that once a subscriber has correctly linked their license in the future, Visual Studio Team Services will automatically apply the best available license to their user after they login the next time. Therefore, for the identity map, you can always specify that a user is given a Stakeholder license that will allow them minimal access to Team Services without needing to pay for a license and then let them fix their subscription linking in the future.

Review identities with a status of NO MATCH

You will need to review all the identity mappings that have a status of `NO MATCH`. This status means that a matching Azure Active Directory identity could not be found.

For additional details on resolving `NO MATCH` to be `OK`, please see the page at https://aka.ms/VSTSIdentityMapNoMatch

5 Get Ready for Import

Create an Azure Storage Container in chosen datacenter

Using the TFS Database Import Service for Visual Studio Team Services requires having an Azure Storage container in a specific Azure datacenter. In most cases, that's the same region as you intend for your Team Services account. For example, if you intend for your Team Services account to be created in the Central United States datacenter, then you will want to create the Azure Storage container in that same datacenter. You can find out where your Azure Storage Container must be located at https://aka.ms/VSTSUploadRegion

☐ **Task:** Create an Azure Storage Container in the same datacenter as the final Visual Studio Team Services account.

For more on creating storage containers, visit https://aka.ms/CreateAzureStorageContainer.

Worksheet

Azure Storage Container Name:	
Datacenter Location:	

Get Ready for Import 5

Set up Azure subscription for billing

A grace period is placed on the newly imported Visual Studio Team Services account to allow your team to finish any steps it needs and correct license assignments. If you anticipate needing to purchase any additional user plans, build/deployment pipelines, hosted build services, hosted load test services, or other developer services, we highly recommend making sure that you have an Azure Subscription ready for linking to your imported Team Services account once the import has completed. The grace period ends on the first day of the following month after you have completed your import.

We will remind you again in Phase 6 for when you will need to do the linking. This preparation step is more about making sure that you know which Azure Subscription you will use in that later step. You can find out more about setting up an Azure Subscription for Visual Studio Team Services billing at https://aka.ms/SetupVSTSBilling.

6 Import

Microsoft

Import 6

☑ Task summary

- ☐ **Dry run of end-to-end import:** Complete a dry run of the end-to-end import before progressing to your production import.
- ☐ **Detach the team project collection:** Detach the team project collection in TFS Administration Console.
- ☐ **Create portable backup:** Create portable backup of the Team Project Collection SQL database.
- ☐ **Upload SQL database backup:** Upload SQL database backup and identity map to Azure Storage Container.
- ☐ **Generate SAS key:** Generate a SAS key for the Azure Storage container and modify your import settings file to include the SAS Key.
- ☐ **Delete previous dry run accounts:** Delete any previous dry run Visual Studio Team Services accounts.
- ☐ **Rename imported account:** Rename the imported Visual Studio Team Services account to the desired name that was reserved in Phase 1.
- ☐ **Set up billing:** Set up the billing for the Visual Studio Team Services Account with the Azure subscription identified in Phase 5.
- ☐ **Reconnect to new account:** Reconnect on-premises build servers to the newly-imported Visual Studio Team Services account.

6 Import

The great news is that your team is now ready to begin the process of starting a dry run of your import and then finally a full production import run. In this phase of the guide, we will walk through the final steps to queue an import as well as discuss the other topics that typically come up at the end of the migration project to get you prepared.

☐ **Task:** Complete a dry run of the end-to-end import before progressing to your production import.

Considerations for roll back planning

A common concern that teams have for the final production run is to think through what the rollback plan will be if there is anything that is wrong with the import itself or if you quickly figure out the import settings provided are not correct including identity map issues like incorrect license or missing team members from the identity map. This is also why we highly recommend doing a dry run to make sure you are able to test the import settings and identity map that you provide to the TFS Database Import Service.

Rollback for the final production run is fairly simple. Before you queue the import, you will be detaching the team project collection from Team Foundation Server which will make it unavailable to your team members. If for any reason, you need to roll back the production run and have Team Foundation Server come back online for your team members, you can simply attach the team project collection on-premises again and inform your team that they will continue to work as normal while your team regroups to understand any potential failures.

Timing worksheet

One thing that is very helpful is to keep track of the timing for each of the steps during the dry run import so that you can start to plan out your final production import timeframe. The worksheet below will help you keep track of the different steps to time for your dry run(s) and production import.

Time for each step

	Dry Run Import #1	Dry Run Import #2	Production Import
Detach Collection			
Generate Backup of SQL Database			
Upload Backup and Identity Map to Azure Storage			
Queue Import			
Final UAT Verification of Imported Team Services Account			

6 Import

Detach your team project collection from Team Foundation Server

Before generating a backup of your SQL database, the TFS Database Import Service requires the collection to be completely detached from Team Foundation Server (not SQL). The detach process in TFS transfers user identity information that is stored outside of the collection database and makes it portable to move to a new TFS server or in this case, to Visual Studio Team Services.

☐ **Task:** Detach the team project collection in TFS Administration Console.

Detaching a collection is easy from the TFS Administration Console on your TFS server. There is a walkthrough for detaching the team project collection at https://aka.ms/DetachTFSCollection.

Generate database backup

For smaller collection databases under 150 GB, the TFS Database Import Service is able to import from a specific SQL backup format: DACPAC. You can find the command-line tool necessary for generating DACPAC files in the SQL Server Data Tools. Here is a sample command-line entry for generating a DACPAC backup file.

```
SqlPackage.exe /sourceconnectionstring:”Data Source=localhost;Initial
Catalog=Tfs_Foo;Integrated Security=True” /targetFile:C:\DACPAC\
Tfs_Foo.dacpac /action:extract /p:ExtractAllTableData=true
/p:IgnoreUserLoginMappings=true /p:IgnorePermissions=true
/p:Storage=Memory
```

There is more information about generating DACPAC backup files and where to find SQL Server Data Tools available at https://aka.ms/CreateTFSBackupDACPAC.

☐ **Task:** Create portable backup of the team project collection SQL database.

Alternate method: importing large collection databases

If your collection database is larger than 150 GB, you will not want to generate a DACPAC backup of your SQL database. There is an alternate method that you will need to take which is setting up your own SQL Server in the same Azure datacenter, restoring the database there, and updating your import settings with a connection string to your database for the TFS Database Import Service to use to create a direct connection for importing your database.

You can find out more about the alternate method of importing if you have a large collection at https://aka.ms/VSTSImportLargeCollection.

Dry run only: attach team project collection again

If this is your **dry run import,** once the SQL database backup of the fully detached TFS team project collection has fully completed, you can attach the team project collection again to make it available to your team members while you continue the rest of the import steps.

If this is your production import, we **do not** recommend attaching your collection to TFS again unless you need to rollback your final import attempt and have TFS available for your team members to continue working.

Upload backup and identity map to Azure Storage Container

Once you have your DACPAC backup file ready, you can upload it to the Azure Storage container that you created in Phase 5 of this guide. You will also want to upload your identity map to the same location. The time to copy can vary depending on your Internet speed and the size of your backup file.

One of the best methods for copying to an Azure Storage container is by using the AzCopy tool. You can find out more how to use it at https://aka.ms/StorageAzCopy.

☐ **Task:** Upload SQL database backup and identity map to Azure Storage Container

Import

Generate SAS key for the Azure Storage Container

The last setting in the import settings file that you will need to update is the SAS key for the Azure Storage Container so that the TFS Database Import Service can securely connect to the storage container to give the Import Service the minimal set of permissions needed to access your team's data. The SAS key can even be time limited to cut off access after a desired time period. It is strongly recommended that you time limit the key to be enabled for at least a minimum of seven days.

Note: It is important to treat the SAS key as a secret. Do not leave the key in an insecure location as it grants read and list access to any data that you have stored in the container.

You can find out how to generate a SAS key at https://aka.ms/GenerateSASKey.

☐ **Task:** Generate a SAS key for the Azure Storage container and modify your import settings file to include the SAS key.

Delete previous dry run import Visual Studio Team Services accounts

Before you can run a second dry run import or the final production import, you will need to make sure you delete any previous Visual Studio Team Services accounts that were created in a previous dry run. You can follow the steps at https://aka.ms/DeleteVSTSAccount.

☐ **Task:** Delete any previous dry run Visual Studio Team Services accounts.

Import 6

Queue the import

You are now ready to queue the import with the TFS Database Import Service. Now that you have your database backup and identity map uploaded and the import settings completed, you can simply queue the import with the following command.

```
TfsMigrator import /importFile:C:\TFSDataImportFiles\import.json
```

This will begin the import and the owner identified in the import settings file will receive an e-mail whenever the import has failed or succeeded. If you have any questions during the import or if you have received a failure, you can contact the team at vstsdataimport@microsoft.com.

Post-import steps

A success e-mail will be sent to the account owner as soon as the import has successfully completed. At this point, anyone with access will be able to login to the newly imported Visual Studio Team Services Account. There are a few remaining items that you may want to perform but for the most part, the Team Services account is ready for your team members to use.

We have captured some of the common steps here but you can find an updated list of post-import topics at https://aka.ms/VSTSPostImport.

Rename final imported account to desired name

In Phase 1 of this guide, you may have preemptively created accounts with the final Team Services account names that you want to use. If this is your final import, you can rename your newly imported Team Services account to that desired name. The steps to take at a high-level are:

1. Rename placeholder account to a different name. For example, from contoso.visualstudio.com to contoso-old.visualstudio.com
2. Wait for a short amount of time
3. Rename the newly imported account to the desired name. For example, from contoso-import.visualstudio.com to contoso.visualstudio.com

You can rename a Visual Studio Team Services account by following the directions at https://aka.ms/RenameVSTSAccount.

☐ **Task:** Rename the imported Visual Studio Team Services account to the desired name that was reserved in Phase 1.

6 Import

Set up billing

Now that the Visual Studio Team Services Account is created, you can complete the final steps of linking the Azure subscription identified in Phase 5 of this guide so that billing is now linked correctly.

Note: If you are not needing to purchase any additional licenses or needing any additional services like builds, deployments, load testing, marketplace extensions, etc. then you can skip this step.

As a reminder, the final steps of setting up billing with an Azure subscription are detailed at https://aka.ms/SetupVSTSBilling.

☐ **Task:** Setup the billing for the Visual Studio Team Services Account with the Azure subscription identified in Phase 5.

Configure build agents

If you were using automated build or deployment servers in your Team Foundation Server environment, you can now connect them to your Visual Studio Team Services account. As part of the import, all of your build definitions have been brought over, but agents and pools need to be reconfigured against the new Team Services account.

You can find the additional steps needed at https://aka.ms/VSTSBuildsPostImport.

☐ **Task:** Reconnect on-premises build servers to the newly imported Visual Studio Team Services account.

Hosted build and deployment pipelines

Your team can also explore taking advantage of the hosted build servers available now that your team has adopted Visual Studio Team Services. Many customers who have transitioned from Team Foundation Server to Visual Studio Team Services have told us that they were able to retire some of their custom build hardware by leveraging the hosted build & deployment services available in Team Services.

Summary

Congratulations! Your team is now in Visual Studio Team Services and you do not need to worry about upgrading your instance again. Now that you are in Team Services you will receive new updates frequently and have many opportunities to implement tools and processes that will help your team to be more effective in building quality software.

Roadmap and release notes for Visual Studio Team Services

New releases are deployed approximately every three weeks and we want to make sure you and your team have the information you need to stay up to date with everything that is new. At the same time, we want to make sure you know where we are investing in helping your development teams.

You can find both our roadmap and a detailed set of release notes for each deployment at https://aka.ms/VSTSFeaturesTimeline.

Deployment roadmap and release notes.

https://aka.ms/VSTSFeaturesTimeline

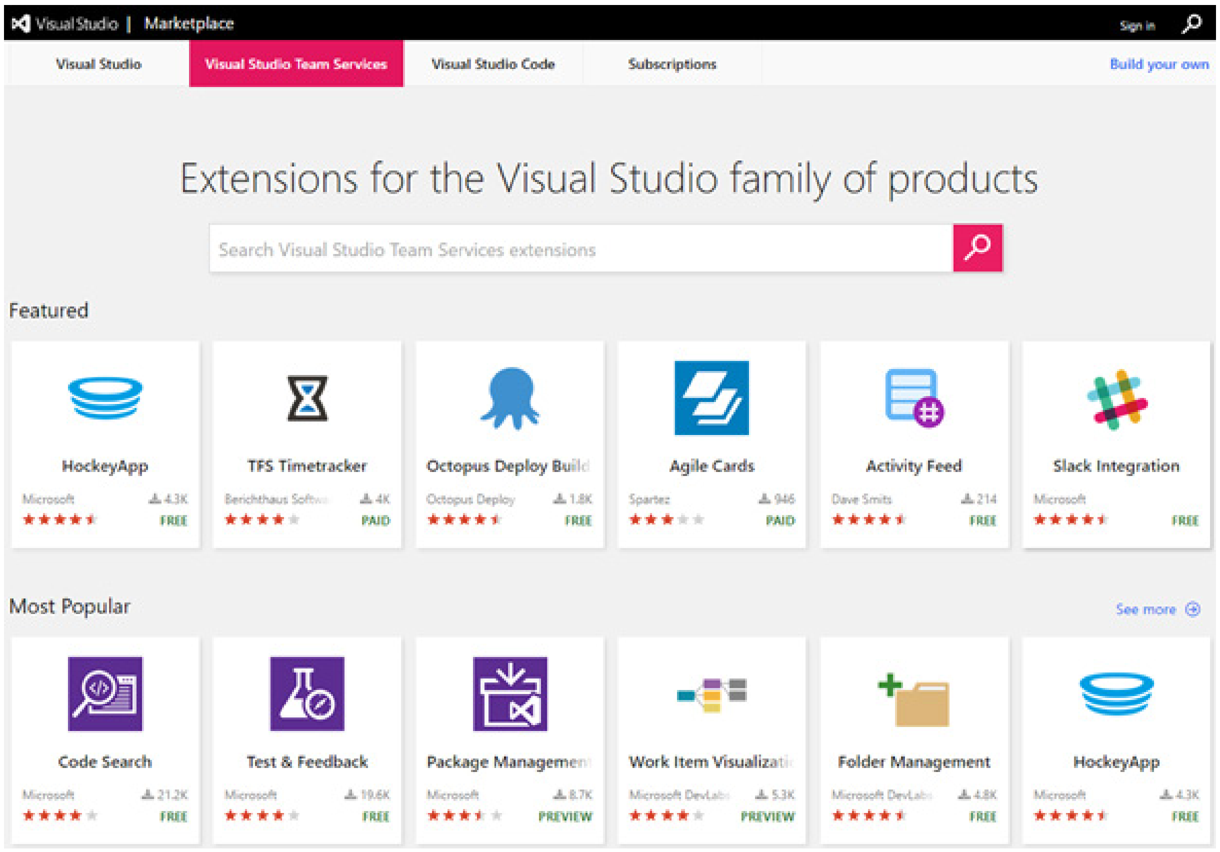
Summary

Find new extensions in the Visual Studio Team Services Marketplace

The Visual Studio Team Services Marketplace is one of the major benefits of adopting Team Services because each of the extensions are extremely easy to install and configure with your Team Services Account. There are many extensions that will add new build/deployment tasks, integrations with many third-party services, and add incredible value for your teams to get more out of the information & functionality in Team Services.

Some of the first extensions you should immediately install are:

- Code Search
- Package Management
- HockeyApp
- Microsoft Teams Integration
- Test & Feedback
- Application Insights


Visual Studio | Marketplace
Sign in
Visual Studio
Visual Studio Team Services
Visual Studio Code
Subscriptions
Build your own
Extensions for the Visual Studio family of products
Search Visual Studio Team Services extensions
Featured
HockeyApp Microsoft 4.3K FREE
TFS Timetracker Berichthaus Softw 4K PAID
Octopus Deploy Build Octopus Deploy 1.8K FREE
Agile Cards Spartez 946 PAID
Activity Feed Dave Smits 214 FREE
Slack Integration Microsoft FREE
Most Popular
See more
Code Search Microsoft 21.2K FREE
Test & Feedback Microsoft 19.6K FREE
Package Management Microsoft 8.7K PREVIEW
Work Item Visualization Microsoft DevLabs 5.3K PREVIEW
Folder Management Microsoft DevLabs 4.8K FREE
HockeyApp Microsoft 4.3K FREE

You can find the Visual Studio Team Services Marketplace at https://aka.ms/VSTSMarketplace.

Summary

Move forward with other developer services from the Microsoft Cloud

Working in Visual Studio Team Services means that it is much easier to take advantage of the many other services available for development teams. Be sure to look at these additional services in the Microsoft Cloud.

- Microsoft Azure
- SQL Server and SQL Azure
- Application Insights
- HockeyApp
- Xamarin Test Cloud
- Development/Test Labs in Microsoft Azure
- Office 365
- Dynamics

Stay connected and involved

As your teams start to leverage more of the many solutions available in Visual Studio Team Services, please be sure to stay connected and involved with helping us know what you think we should be building to make Team Services even better. We have a User Voice site where you can suggest new features and vote on other developer's feature requests to help us prioritize our future investments. You can find that User Voice site at https://aka.ms/VSTSUserVoice.