brief contents

PART 1 EXPLORING TEAM FOUNDATION SERVER.........................1
  1 • TFS and the practice of software development  3
  2 • Exploring the changes in TFS 2008  16
  3 • Introducing VSTS 2008 Database Edition  48

PART 2 DIVING DEEP INTO VERSION CONTROL AND TEAM BUILD.85
  4 • Understanding branching in version control  87
  5 • Understanding branch policies  119
  6 • Understanding merging in version control  141
  7 • Understanding Team Build  160
  8 • Versioning assemblies using Team Build  205

PART 3 ADMINISTERING AND CUSTOMIZING TFS.......................229
  9 • Configuring and extending TFS  231
 10 • Using workflow with TFS  268
In part 1 of the book, we take a broad look at some of the salient features of Team Foundation Server (TFS). We learn how TFS fits into the overall application lifecycle management (ALM) process, what the major features are, how the platform is architected, and how it can be extended.

We also review the changes introduced in TFS 2008—such as major enhancements in Team Build, incremental improvements in Team Foundation version control, and integration of Visual Studio Team System 2008 Database Edition.

Since our goal is to obtain a general understanding of how TFS works, the material in part 1 is mostly introductory in nature. We cover a lot of ground and aim to develop a broad understanding of the major functionalities in TFS.
Visual Studio Team System 2008 (VSTS) is Microsoft’s application lifecycle management (ALM) platform for managing the end-to-end software development process. It’s a vast product with wide-ranging capabilities. VSTS first debuted in 2005, with incremental improvements added in 2008, and contains both general purpose as well as role-based capabilities. The platform can be extended as needed. Given how geographically distributed, technically complex, and strategically important enterprise software projects have become, I feel that VSTS deserves a careful look with regard to solving real-life development challenges. The days are gone when you could hope to develop enterprise software, with acceptable quality and within a reasonable timeframe, without effective tool support.

This book is about improving your software projects by leveraging the capabilities of the Microsoft ALM platform. In this book, we learn how to resolve various

This chapter covers

- Evolution of software development practices
- TFS architecture
- High-level functionalities
practical issues by applying the system’s native capabilities, as well as by augmenting them via customizations.

As practitioners in the information technology (IT) industry, we all know that software development isn’t easy. Why this is so has been extensively debated since the industry was born. Despite the advent of myriad development methodologies, infrastructure tools, and careful project management, most projects still run behind schedule and over budget. A recent study showed that only 35 percent of software projects can be categorized as successful (completed on time, within budget, and met user requirements).

Of course, this situation isn’t unique to the IT industry. We see unexpected delays and cost overruns when large-scale projects are undertaken in many other sectors, such as construction, transportation, and life sciences. Perhaps the complex and volatile nature of the real world makes it impossible for humans to make perfect predictions. Nevertheless, unless we constantly strive to improve our estimation and execution processes—with the understanding that perfection is a goal, not a yardstick to beat people with—progress, however incremental, will become impossible. Chaos comes easy.

While I don’t know of an absolute “right” way to do software development, I do feel that outcomes can be improved by focusing on the basics. Industry studies have shown that key success factors include executive support, effective communication between stakeholders, good architecture, skilled team members, incremental development, extensive testing, requirement stability, and the ability to dynamically adjust course based on changing realities. There are no surprises here. Therefore, the question becomes not so much whether we know what the best practices are as much as how best to practice them on a day-to-day basis. Of course, you need to deal with some issues (such as obtaining and retaining skilled resources) in an administrative manner. For others, it’s critical that you select an ALM platform that can mitigate execution risks and promote adoption of good practices. VSTS is such a platform.

The client portion of VSTS is segmented along various roles—architects, developers, testers, and database professionals (see figure 1.1). If you’re interested in all client-side features, you can purchase Visual Studio Team System 2008 Team Suite, which combines the full range of capabilities. Stakeholders who need to participate in the lifecycle but don’t want to install the Visual Studio IDE (integrated development environment) can use the Visual Studio Team System Web Access 2008 Power Tool (TSWA). TSWA enables you to access TFS using only a web browser.

Visual Studio Team System 2008 Team Foundation Server (TFS) is the server-based portion of VSTS. TFS offers core features such as process definition and instantiation, work item tracking, source code management, project portal, build management, reporting, and so on.

In this chapter, we review the high-level features of TFS. You’ll learn about the following:
Background on software development processes

1.1 Background on software development processes

The practice of software development has evolved considerably in the last 50 years. When early computer programs were written in the 1950s and 1960s, engineers mostly followed a code and fix approach to solve business problems. This ad hoc process soon became untenable for large-scale development projects. In the 1970s, the waterfall model, emphasizing up-front design and meticulous planning, was proposed as a way to reduce execution risk. The waterfall model had an enormous impact on software engineering and continues to be followed in many organizations today.

In the 1990s, the IT industry went through seismic changes. The primary drivers were business-as well as technology-related. With the introduction of personal computers in the 1980s, computing power increased dramatically, while the associated hardware and software costs went down. Global communication was revolutionized with the advent of the World Wide Web in the 1990s. Businesses became geographically unfettered.
and 24/7 service availability became routine. The new world demanded rapid software
delivery in the face of constantly changing business realities.

As a result of these transformations in the 1990s, software projects increased in
complexity while simultaneously demanding drastic cuts in implementation time. The
requirements for security, availability, interoperability, scalability, and performance
became much greater than they were in the 1970s or 1980s. Furthermore, develop-
ment teams became distributed globally and started working asynchronously—signifi-
cantly increasing the chances of miscommunication and mismanagement.

Beyond the waterfall, new software development methodologies were proposed to
meet the requirements of the brave new world. Iterative processes, emphasizing incre-
mental delivery of value, were proposed in the 1990s. Iterative processes such as the
Microsoft Solutions Framework (MSF), Rational Unified Process (RUP), and others
essentially suggested mini waterfall-based cycles—the development period was divided
into iterations that incrementally enriched a core set of key functionalities.

In early 2000, agile processes were proposed in an attempt to further reduce devel-
opment time and increase productivity. Agile processes, such as extreme program-
ming (XP), Scrum, Crystal, and feature-driven development (FDD), advocate high-
 fidelity communication (mostly face to face). They also significantly reduce documen-
tation, eliminate big up-front planning and design work, advocate testing as an inte-
grated part of development, and relentlessly emphasize creating working software that
meets immediate needs.

However, despite the prevalence of so many methodologies, the industry-wide suc-
cess rate for software projects remains low. Why has the project success rate not
improved in the last decade? We’ve already discussed many reasons, but in my opin-
ion, one of the main reasons has been the lack of widely available, cost-effective, and
integrated toolsets to implement a chosen process on an end-to-end basis. Although, in
theory, a process may sound great, if you don’t have the right infrastructure to realize
it, the discussion remains largely academic.

Microsoft’s entry in the ALM market opens up new possibilities. The Microsoft
Solutions Framework (MSF) provides a process framework where you can plug in your
chosen process. VSTS ships with two built-in process templates—MSF for Agile Soft-
ware Development and MSF for CMMI Process Improvement. You can add other third-
party process templates (such as RUP or Scrum) or create a custom process.

TFS provides the infrastructure needed to support enterprise-wide processes across
distributed organizations. The artifacts, policies, and workflows defined in the process
are available to the entire team via TFS. TFS also provides the metrics and reports
needed to gain visibility regarding project status, trends, and roadblocks. The ultimate
goal is to provide a “friction-free” environment where the infrastructure enhances col-
 laboration, productivity, and quality.

1.2 **TFS architecture**

TFS is based on a distributed architecture. The product consists of three logical tiers
(see figure 1.2). They are discussed in sections 1.2.1 through 1.2.3.
1.2.1 Client tier

The client tier consists of components that interact with the user. It includes Team Explorer, add-ins for Microsoft Excel and Microsoft Project, command-line tools, and third-party VSIP (Visual Studio Industry Partners) packages. Components in the client tier communicate with the application tier via a set of APIs defined in the Team Foundation object model.

1.2.2 Application tier

The application tier consists of a number of web services that provide access to core system services. The services are grouped in two categories—data services (work item tracking, source control, build management) and integration services (security, registration, events, linking, classification). Additionally, the application tier hosts the SharePoint-based portal and SSRS (SQL Server Reporting Services)–based reports.

1.2.3 Data tier

The data tier consists of operational databases, a data warehouse, and an Analysis Services (OLAP) cube. This tier stores persistent data related to work items, source code, builds, team projects, security, activities, notifications, and so on.
1.3 **Major features**

The features offered by TFS can be divided in five categories (see figure 1.3). They are discussed in sections 1.3.1 through 1.3.5.

1.3.1 **Work item tracking**

The work item tracking (WIT) system enables you to create, modify, copy, and query work items. In TFS, the term *work item* denotes an item that you want to track during the development process. Work items include bugs, tasks, scenarios, requirements, change requests, and so on. The work item types that are available in a team project are defined in the corresponding process template. You can also create custom work item types in addition to the built-in ones.

A work item type definition contains field definitions, form layout information, and state transition rules. Work item type definitions are stored in XML files associated with process templates. You can use the Process Template Editor included in the TFS Power Tools to graphically edit the work item types (download TFS Power Tools from http://msdn2.microsoft.com/en-us/tfs2008/bb980963.aspx).

Work items can be queried using a SQL-like query syntax called *Work Item Query Language (WIQL)*. The VSTS IDE includes a graphical query editor for creating and executing queries. You can also create and execute WIQL queries from custom program code using the TFS object model.

The work item form (see figure 1.4) is fully customizable. You can change the look and feel by creating new tabs, adding or removing form fields, and inserting custom...
controls. Work items can be linked to one another (support for hierarchically linked work items is coming in the next version), can contain attachments, and also have an audit log showing the change history.

Work items can be bidirectionally synchronized with Excel and Project for consumption by stakeholders who may use these products for data entry and project management. You can also use the TFS object model to synchronize work items with other products as needed (such as your enterprise project management system).

1.3.2 Version control

Team Foundation version control (TFVC) provides the capability to store files in a central repository, along with their version information (see figure 1.5). Effective support for managing and evolving source files is critical for implementing an ALM platform that helps, not hinders, the development process. Arguably, the most important asset in your development effort is the source code itself. You need a flexible and reliable system for developers to store code files, secure them, retrieve them as needed, isolate concurrent development efforts, and attach metadata to each check-in.

TFVC provides reliable change management via atomic check-ins. When you check in a set of changes as part of a single changeset, either all changes will be committed to the repository or none will be committed. This transactional check-in behavior ensures that the codebase remains in a consistent state at all times. You don’t have to worry about a partial update breaking the internal consistency of the codebase.

TFVC supports check-in policies that help protect the integrity of the codebase. Check-in policies help ensure that the code being checked in meets appropriate quality criteria. The quality criteria may include passing static code analysis, associating check-ins with work items, and so on. You can also create custom check-in policies and implement your own change control criteria. If a developer checks in code by overriding a policy, TFVC can send out an email alert containing the reason. The policy override information is also recorded in the data warehouse, allowing the creation of reports to track the action.

![Figure 1.5](image)

Figure 1.5 Version control provides a central repository for source code and other artifacts.
TFVC makes it possible for developers to create shelvesets. Shelvesets are used to store pending version control changes, associated work-item links, and comments on the server without performing a check-in operation. Developers can use shelvesets to back up changes before starting a new task, to easily share changes between multiple developers, or to perform code reviews.

TFVC offers low-overhead branching and merging capabilities to facilitate parallel development. Branching allows a collection of files to evolve on two independent paths, or branches. Branching creates a parent-child relationship between the two collections of files, such that changes made on one side of the relationship don’t affect the other until an effort is made to merge the changes from one branch to another. Branching is commonly used at various milestones within the development lifecycle. During the development phase, multiple child branches can be used to allow teams to develop features in isolation before merging back to a common parent branch. When the product ships, a “release” branch can be created so that product maintenance is done in isolation from the next version’s development. After release, branching can be used to create service packs and patches to support the product.

Merging is used to integrate code between branches. TFVC enables you to easily merge code that’s located in related branches (branches that are related via parent-child relationships). Although less straightforward, TFVC allows you to merge code between unrelated branches using a baseless merge. Typically, all changesets up to a particular point are included in a merge. However, TFVC also supports cherry pick merges, where you can specify certain changesets to be included in the merge. You can also view merge candidates, review merge histories, detect and resolve merge conflicts, and roll back merges.

When you put it all together, you’ll find that TFVC provides a scalable enterprise-class infrastructure for supporting your code management needs. Using third-party products, you can use TFVC within other IDEs and operating systems. Whether you follow an agile or a more formal development methodology, you’ll find that TFVC provides strong support during all phases of the software lifecycle—including development, testing, release, and maintenance.

1.3.3 Build automation

An automated build system that empowers you to build early and often is crucial for success. In addition to simply compiling the source files, a good build system generates tracking and quality information for each build. The tracking information should include which files were changed, which bugs were fixed, which features were implemented, who launched the build, and so on. The quality information should include which build verification tests (BVTs) ran successfully, what kind of code coverage was achieved, and the like.

Team Build provides an out-of-the-box integrated infrastructure for managing automated builds. Under the covers, Team Build uses the MSBuild engine to generate
assemblies. However, Team Build adds a host of other features that facilitate traceability, collaboration, and quality. For example, Team Build generates a unique number for each build, downloads source code from TFVC, associates appropriate work items with the build, performs static code analysis, executes BVTs, generates code coverage information, and pushes the generated binaries to designated drop locations. You also get a build report showing the changesets that were included in the build, the work items that were resolved, relevant quality and timing information, and other metadata associated with the build (see figure 1.6).

You can customize the build process in several ways. You can create custom MSBuild tasks that can fine-tune the build process using the full power of your chosen programming language. These tasks can be invoked at specific points during the build process. Team Build specifically provides customizable targets that you can override for extending the build process. Additionally, Team Build provides a number of properties that you can configure to customize the build process.

In addition to streamlining the build execution process, Team Build provides a number of features that facilitate build management. You can specify which kinds of binaries to retain in the drop location and for how long, when to trigger builds (Team Build supports continuous integration builds and rolling builds as well as scheduled builds), which build machine to use for executing builds, which directories to use for downloading the source files and for generating the output binaries, and so on. Appropriate team members can also indicate the quality and QA status of each build (such as Initial Test Passed, UAT Passed, Ready for Deployment, and so forth). Team Build can also generate notifications when a build is completed or when a build quality is changed.

Figure 1.6 Team Build offers a full-featured build creation and management infrastructure.
1.3.4 Project portal

Effective communication between stakeholders is a key enabler of success in any software development effort. Without effective communication, requirements get misconstrued, priorities get misaligned, and resources get misplaced. When working with distributed or offshore teams, the need for effective communication is magnified many times. In any software project, you must ensure smooth information flow and augment it with effective metrics.

TFS facilitates information exchange by creating a SharePoint-based project portal for each team project (see figure 1.7). The project portal serves as the focal point for all stakeholders, across departmental boundaries. The portal hosts shared documents, reports, and key performance indicators (KPIs), threaded discussions, announcements, process guidance information, meeting workspaces, custom links, and whatever else you might consider appropriate for promoting collaboration. Anybody can access the project portal and obtain relevant information using just a web browser from anywhere in the world—without installing Visual Studio, acquiring VSTS licenses, or worrying about firewalls. In addition to offering timely information, the portal provides a sense of community, continuity, and context. The project portal can also serve as an executive dashboard for members of the executive team.

TFS supports project portals based on Windows SharePoint Services 3.0, Microsoft Office SharePoint Server 2007, and Windows SharePoint Services 2.0. The default project portal is created during the creation of a team project. You can customize the portal using the standard customization options available in SharePoint.

Figure 1.7 The Project portal promotes collaboration and transparency.
Major features

You can change the look and feel of the portal using SharePoint Designer, as well as create custom web parts and add them to the portal. You can create custom reports and perform additional analysis. You can also create custom workflows associated with shared documents. The default document libraries and their contents can be customized by modifying the WssTasks.xml file associated with the process template. The site template itself can also be customized so that new portals are based on your custom design.

1.3.5 Reporting

Reports enable you to keep your finger on the pulse of your development project. You can view information in near–real-time regarding critical aspects of project health—bug rates, velocity, test results, regressions, remaining work, build results, work items, and so on (see figure 1.8). Additionally, you can create custom reports and KPIs to better understand progress, identify bottlenecks, detect trends, and predict outcomes. Effective project management is impossible unless you have access to accurate data and can easily analyze the information using your preferred representational formats (views).

TFS reports are based on SQL Server Reporting Services (SSRS). You can use the included reports out of the box, modify them, or create new ones using Visual Studio Report Designer. Your custom reports can be included in the process template so that the new reports are automatically included in new team projects (which are created based on the modified process template). For wide distribution, reports can be placed...
in the SharePoint-based project portal, in addition to being accessible from Team Explorer. For ad hoc reporting purposes, you can use Excel (or other third-party tools) to slice and dice the underlying data.

Reports access the data stored in back-end relational and Analysis Services databases. The TFS data repository consists of a number of operational databases (not used for reporting), a relational data warehouse, and an Analysis Services database (OLAP cube). At regular intervals, the content and the schema of the data warehouse as well as the cube are refreshed based on the data in the operational databases. This approach provides flexibility and performance.

The Analysis Services database contains a cube named Team System that is most often used in reports to display aggregate information. The cube contains several measure groups (such as code churn, test result, and current work item) and dimensions (such as date, team project, and build quality). For ease of navigation, and to provide focused context, the cube is logically subdivided into a number of categories called perspectives (such as build, current work item, and test result). Like any other Analysis Services cube, you can enhance the Team System cube as needed to meet your unique needs. For example, you can add KPIs, change the format of the cube, partition the data, and so on.

1.4 Summary

In this chapter, we learned how TFS fits within the general context of software engineering. We reviewed the platform’s high-level architecture and studied its major subsystems. We saw that TFS enables you to implement your chosen development methodology on an enterprise scale. The product provides a robust ALM infrastructure, consisting of work item tracking, version control, build automation, project portal, and reporting. TFS also interoperates with other products in the enterprise ecosystem, such as third-party requirement gathering tools (for example, CaliberRM), third-party IDEs (such as Eclipse), as well as Microsoft Office products. You can customize TFS as well as extend its out-of-the-box features using the object model APIs. You can also listen to TFS events, and leverage other extensibility points.

In the next chapter, we review the specific improvements that have taken place in TFS 2008. Despite an impressive array of features in TFS 2005, there was room for improvement, especially in the build subsystem. We review the changes and show you how to take advantage of the enhancements.

The rest of the book drills down into real-life issues that arise when applying TFS in the field. As practitioners in the industry, we know that there can be no one-size-fits-all approach. In order to be able to successfully use any horizontal ALM platform within the enterprise, we must have the assurance that the system meets the general needs of security, reliability, scalability, auditability, interoperability, and performance. Not only that, but it also must expose well-designed extensibility points that can be utilized to meet the unique needs of the organization. Furthermore, the platform should have a familiar user interface as well as internal consistency to facilitate easy adoption.
Lastly, the product should be a well-behaved enterprise citizen, for painless hosting, operation, and maintenance in enterprise IT facilities. This implies that the platform should leverage as many existing IT assets as possible (such as databases, portals, reporting services, and so on) for ease of administration as well as to lower cost of ownership. TFS meets many of these enterprise requirements.

Although there’s definite room for improvement in TFS (and we study some of those aspects in detail in the following chapters), we believe that TFS offers a compelling solution for mitigating many common lifecycle concerns. You can start using TFS today to improve your software development process.

1.5 References

TEAM FOUNDATION SERVER 2008 IN ACTION
Jamil Azher

In a project with dozens of developers, hundreds of builds, and thousands of opportunities for mistakes, managing the process is as important as writing the code. Team Foundation Server 2008 is Microsoft’s application lifecycle management (ALM) tool. Using it, you can effectively track work items, create test cases, manage source files, and generate builds. Ramp-up time is minimal because it’s fully integrated into Visual Studio.

Team Foundation Server 2008 in Action shows developers and managers how to collaborate on complex software projects using TFS 2008. You’ll follow real-life scenarios as you learn how to support your preferred development methodology. The techniques are actionable and the solutions are experience-based. You’ll master the out-of-the-box functionalities in TFS as well as learn to customize its source code management, database development, build, and reporting capabilities.

What’s Inside
- TFS for database professionals
- Customizations and workarounds
- Practical tips and tricks

About the Author
Jamil Azher is a technology architect at Microsoft with 18 years of experience in programming, global project management, and entrepreneurial ventures. Jamil earned a Master’s degree in IT from Harvard and a BS degree in engineering from Caltech.

For online access to the author, code samples, and a free ebook for owners of this book, go to www.manning.com/TeamFoundationServer2008inAction

“A must-have book for anyone who wants to learn TFS.”
—Robert Horvick, Microsoft

“Tips and tricks only experience can provide!”
—Eric Swanson
Enterprise Architecture Consultant

“An impressive toolkit for mastering Team Foundation Server.”
—David Barkol
Author of ASP.NET AJAX in Action

“Clear, distilled, nuggets of advice based on real-world experience.”
—Benjamin Day
Benjamin Day Consulting, Inc.