PreEmptive Analytics for Team Foundation Server

Quick Start Guide

Contents

Overview

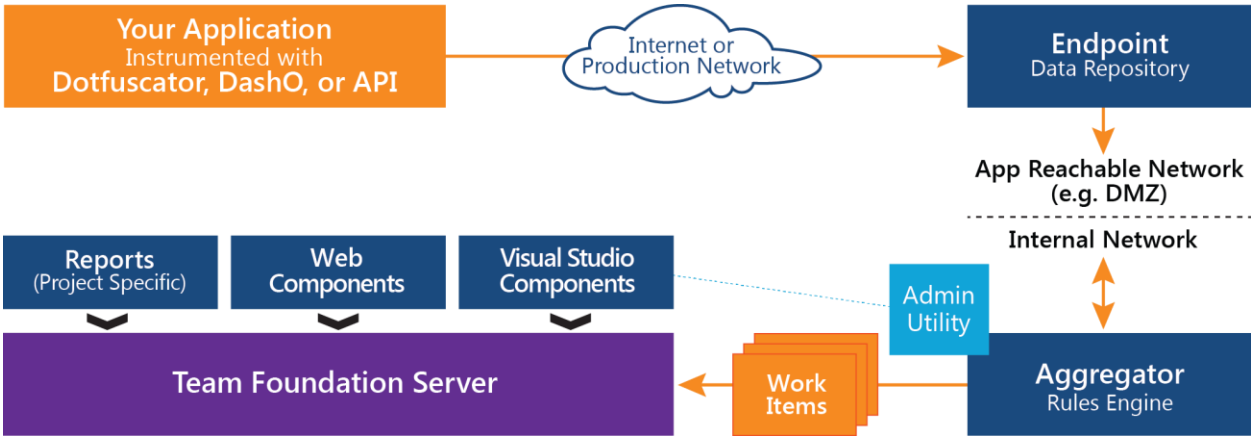
PreEmptive Analytics for Team Foundation Server (PA/TFS) allows you to see, as they happen, the errors your users are experiencing. By aggregating and analyzing exceptions reported by your production applications and automatically creating TFS work items, PA/TFS gives you access to unprecedented levels of incident data right where you need it – in Visual Studio, TFS Web, and Reporting Services.

This document will help you install and configure PA/TFS Professional in a production or production-like environment. If you are using PA/TFS Community Edition (CE), please follow the instructions in Microsoft's Hands-On Lab for PA/TFS instead. If you are installing Pro but plan to just install it on a single machine for test purposes, please use the "Basic" mode of the installer and ignore the parts of this document that describe how to install PA/TFS on multiple machines.

Note: *To use PA/TFS, you must also create an instrumented application and configure it to send exception data to the PA/TFS environment you are about to install. Please see Instrumentation and Obfuscation Quick Start (Dotfuscator) for instructions, or contact Support for assistance. If you have not already obtained a copy of Dotfuscator, DashO, or one of our instrumentation APIs, please contact Sales.*

Component Overview

PA/TFS is made up of five individual components that are distributed via two installers. Those components are shown in dark blue in the diagram below.



- **Endpoint:** Provides a web service that accepts exception messages and stores them for evaluation.
- **Aggregator:** Periodically queries the Endpoint and evaluates new data against the configured rules. Configured by a machine-local configuration (admin) utility, and/or by a project-specific (limited) version of the utility installed as part of the Visual Studio Components.
- **Visual Studio Components:** Provides a Visual Studio control for viewing stack traces and incident details. Provides a custom Team Explorer pane for accessing the work items created by PA/TFS.

Provides a lightweight version of the configuration (admin) utility for configuring project-specific settings.

- **Web Components:** Provides a control for viewing stack traces and incident details in TFS Web.
- **Reports:** Provides reports (for each TFS project) that summarize the work items created by PA/TFS for that project.

The Endpoint and Aggregator are distributed via the **PreEmptive.Analytics.Pro.Setup.exe** installer. The Visual Studio Components are distributed via the **PreEmptive.Analytics.VisualStudio.exe** installer. The Web Components are installed alongside the Aggregator and must be manually installed in TFS Web using the instructions below. The Reports are pushed into Reporting Services by the configuration (admin) utility for each TFS project that is configured to use PA/TFS.

Installation Best Practices

Please see the PA for TFS User Guide for detailed installation instructions. This document only provides an overview to help you get started.

Endpoint and Aggregator

The Endpoint and Aggregator together provide the core feature of PA/TFS: accepting exception messages, processing those messages through the configured rules, and creating TFS work items based on the rule results. That feature is split across two separate runtime components in order to support different network topologies, especially corporate networks that are split into a DMZ and an internal network.

The **Endpoint** is designed to run in a DMZ; it is a small web service that only accepts messages and stores them in a database. It never sends/pushes that data anywhere else, so it does not require a firewall hole to reach the internal network. It can respond to queries (i.e. from the Aggregator) to deliver new messages to the internal network upon request. Note that it is not necessary to run it in a DMZ; it can be run on the same machine as the Aggregator if that meets your needs.

The Endpoint must be network-reachable by your production instrumented applications. Its host/URL should be long-lived and stable, because it will be hard-coded into your instrumented production applications.

The Endpoint requires IIS and access to a SQL Server (or SQL Express) database.

The **Aggregator** is designed to run inside the internal network. It is a Windows service that periodically (60 seconds by default) queries the Endpoint for new data, then processes that data through the configured rules, and then creates or updates work items in TFS as needed. The Aggregator is also responsible for configuring individual TFS projects to use PA/TFS. It creates a local time-limited cache of the data it processes, and also caches some state-tracking data in TFS as work item attachments.

The Aggregator must have network access (for a web service call) to the Endpoint, must have network access to the TFS server, and must run as a user that has sufficient rights to configure TFS team projects

and create and update work items in those projects. The Aggregator is also able to query and update the TFS Maximum Attachment Size setting (as a user convenience), but only if it is running as a user with TFS administrative rights.

We recommend starting with the Endpoint and Aggregator on separate hosts. If any of your applications – now or in the future – run outside your network, the Endpoint should be in the DMZ. If not, the Endpoint can be installed inside the internal network. In either case, it should have a stable hostname/URL. The Aggregator can usually be installed on the same host as TFS, but if you have scalability or performance concerns you may want to install it on its own machine.

There are certain scenarios that might require multiple Endpoint instances and/or multiple Aggregator instances. These scenarios should be very rare; please contact Support if you believe you might need such a topology.

Visual Studio Components

The Visual Studio Components provide custom controls that are necessary to view the work items (incidents) created by PA/TFS from within Visual Studio. They also provide convenience features for accessing those work items and for viewing and editing project-specific rules and settings.

The Visual Studio Components must be installed in every instance of Visual Studio that will be used to access work items created by PA/TFS.

Note that Visual Studio 2012 comes pre-installed with the PA/TFS Visual Studio Components from the Community Edition (CE) of PA/TFS. Installing the Pro version will uninstall the CE version automatically. Please ensure you do so, to avoid version conflicts between PA/TFS CE and Pro.

Web Components

The Web Components provide a custom control that is necessary to view PA/TFS-created work items from within TFS Web. The Aggregator installer will put the Web Components on that same machine, but from there they must be manually installed into TFS Web. (This is a limitation of TFS Web, not of PA/TFS.) The configuration (admin) utility provides a link to the binaries and installation instructions.

The Web Components must be installed in TFS Web if you plan to use TFS Web to access work items created by PA/TFS.

Reports

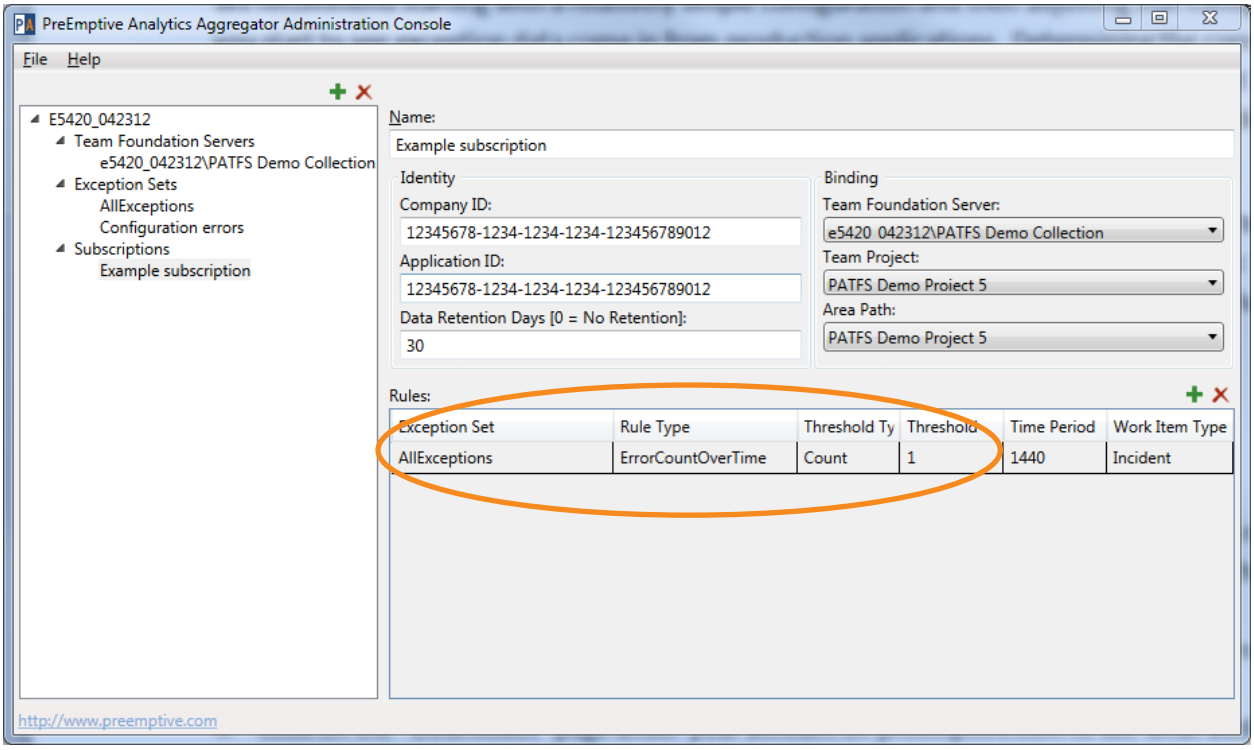
The PA/TFS reports do not have to be administratively installed. They will be deployed to Reporting Services automatically each time the configuration (admin) utility is used to configure a team project to work with PA/TFS, so long as the TFS instance is configured to use Reporting Services.

Configuration Best Practices

Please see the PA for TFS User Guide for detailed configuration instructions. This document only provides high-level advice about what settings might be appropriate in various situations.

We recommend starting with a relatively simple configuration and then adjusting that configuration as you see exception data come in from production applications. Determining the correct settings is primarily a matter of "tuning" the rules to achieve the outcome you want, and it is easiest if you start with very permissive rules (i.e. that create work items for every incident).

To get started, we suggest leaving the default "All Exceptions" exception set. Then create a single subscription for each application you have instrumented, to tie that application to a specific TFS team project. Then modify the default rule to have a threshold of "1" and save it that way until you begin to see production data come in.



From there, you have many options (per subscription):

1. Increase the threshold and adjust the time period of the default (ErrorCountOverTime) rule.
2. Switch the default rule to the TopErrorsOverTime rule type, and tune that rule. (This is especially useful if you have a high volume of exceptions.)
3. Create additional exception sets that watch for specific exceptions, and add additional rules to the subscription to fine-tune how individual exception classes are processed.
4. Look on the Downloads page on preemptive.com for new rule types that might be useful for you.
5. Create your own custom rule to suit your particular need. Please contact us to learn more.
6. Any combination of the above.

Instrumentation Best Practices

Please see Instrumentation and Obfuscation Quick Start (Dotfuscator) for detailed instrumentation instructions. This document only provides advice on PA/TFS-specific issues.

When instrumenting an application for PA/TFS, you do not need to use a PreEmptive-supplied CompanyKey. (That is only necessary if you are using our Runtime Intelligence Service.) You can generate your own arbitrary GUID by using the GUID-generator button on the Guid row under the ApplicationAttribute (in Dotfuscator) and copying the value over to the BusinessAttribute.

Please note, however, that once you instrument an application with a specific CompanyKey, you should reuse that same CompanyKey for all your applications. That will make it easier for you to configure PA/TFS and will make it possible for you to (later) begin using the Runtime Intelligence Service and see all your applications' data under a single account.

It is possible to instrument your applications to send non-exception data (e.g. platform, feature, performance, etc.) but that data will not be processed by PA/TFS. The additional data is only useful if you are using Runtime Intelligence Service.

Possible Surprises

There are two aspects of the design of PA/TFS that might result in unexpected (but intentional) behavior. These aspects are described in the sections below.

Historical Data

PA/TFS is designed around a "subscription" model for processing data. What that means is that, in most cases, new subscriptions and new rules will only process data that is received after the subscription or rule is created. Any previously-received data will not be re-evaluated when a new subscription or rule is created.

On the other hand, once a rule is active, it will cache any data that matches its settings and use that data as needed to correctly perform its function. The historical data cache is preserved as far back as the "Data Retention Days" field specifies, but each rule only looks as far back as its "Time Period," which is usually less.

Work Item Uniqueness

A key feature of PA/TFS is that once a work item is created for a given exception, new instances of that same exception will simply update the prior work item rather than creating a new work item. This generally works as users expect but there are some scenarios that may be surprising.

The key to the issue lies in defining the term "same exception" – i.e. how does PA/TFS decide whether a new exception report matches an old one? Some of the fields it uses can cause surprising side effects:

- **Stack Trace:** our intuitive understanding of whether two exceptions are the "same" would suggest that the stack trace is a critical part of that sameness. However, it is possible for a single line of code

to cause the same exception type to be thrown (e.g. NullReferenceException) even though it was called from multiple sources. Each of those sources would represent a different stack trace so PA/TFS will treat those as different work items. In most cases this is the desired behavior but there are some situations where it might be preferable to create a single work item for all the different stack traces. PA/TFS has no way of identifying those cases, though, so it uses the more-informative option.

- **Application Version:** PA/TFS does not create a new work item for each distinct version of your application that generates the same exception. Instead, it creates a single work item and updates it with counts from all versions, assuming the stack traces remain identical across those versions. This may be undesirable if you fix a bug and then it crops up again (with the exact same stack trace) – no new work item will be created. PA/TFS will, however, update the Version field in the work item to reflect the version reported by each new exception report.
- **Rule ID:** work items are tied explicitly to the specific rule that created them. That has two possibly-unexpected consequences.
  - First, if two rules both match the same exception, and both meet their threshold (at the same time or in sequence), two work items will be created. It will be clear from the work item titles that they came from two different rules but it may still be undesirable behavior. Depending on the rule configuration, though, the presence of multiple work items for the same exception could indicate that the exception is especially severe.
  - Second, if a rule is deleted, any work items that were created by that rule will stop being updated. Recreating an exact copy of that rule will actually create new copies of each of the work items (as new exceptions come in and the thresholds are reached again) and begin updating those new work items from then forward.

These behaviors may be changed in future versions of PA/TFS.