System Optimization Guide

AgilePoint BPMS v5.0 R2 SP1

Document Revision r5.2.15

August 2014

Contents

Preface 5
- Disclaimer of Warranty 5
- Copyright 5
- Trademarks 5
- Government Rights Legend 5
- Virus-free software policy 5
- Document Revision Numbers 5
- AgilePoint Documentation in PDF and HTML 6
- Opening the Documentation Library 6
- Finding Information in the Documentation Library 7
- Downloading Files and Sharing Links from the Documentation Library 7
- Contacting AgilePoint Sales 8
- Contacting Customer Support 8

Scalability 9
- Scale Up 9
- Scale Out 9

Capacity of the Machines in your Environment 10
- CPU Speed 10
- Number of CPUs 10
- CPU Sizing 10
- System Memory (RAM) 10
- Memory Utilization 11
- Hard Disk Performance 11
- Network Bandwidth 11
- Network Adapters 11

Virtual Environments 12
- Development in Virtual Environments 12
- Support for Virtual Environments 12

Dedicated Servers 14
- Non-Dedicated Server 14

Clustered Servers 15
- Clustering Terminology 15
- NLB Cluster 15
  - Scalability 15
  - High Availability 16
- AgilePoint CSM 16

How the AgilePoint CSM Works ... 16
Flow of Events in a NLB Deployment ... 16

General Performance Considerations ... 18
System Usage ... 18
Concurrent Users ... 18
Process Steps Per Day ... 18
External System Load ... 19
Data Metrics ... 19
Performance Bottlenecks ... 19
Network Bandwidth ... 19

AgilePoint Server Performance Characteristics ... 21
Event Driven Architecture ... 21
Time-Sharing Model ... 21
Interruption Handling ... 21
Predicting AgilePoint Server Resource Usage ... 22

Performance Calculations ... 23
Calculating Average Event Processing Time ... 23
How a Faster Workflow Engine Can Increase Capacity ... 23
Scenario to Consider ... 24

Performance Tracing Utility ... 26

Performance Testing and Benchmarking ... 29
Stress Testing ... 29
Benchmarking ... 29

AgilePoint Server Configuration Settings ... 31
Configuring the AgilePoint Server Thread Pool Size ... 31
Thread Pool Sizing Guidelines ... 32
Increasing Thread Pool Size ... 32
Decreasing Thread Pool Size ... 33
Configuring the Swap Out Time ... 33

Database Sizing and Configuration ... 35
Database Sizing ... 35
Database Archiving ... 36
APADM Command Line Utility ... 37
APADM Archive Database Command ... 37
Database Connection Pool Size ... 37

Client Applications ... 39

Tips for Improving Performance and Troubleshooting.....40
Out of Memory Errors..... 40
Virtual Memory.....41
Affect of Customizations on Server Resources and Performance.....41
Hard Disk Fragmentation..... 41
Scheduled Tasks, Screen Savers, etc.....41
File Locking, Indexing, and Scans.....42
Anti-Virus Software, Firewalls, Pop-up Blockers, Proxies, etc.....42
AgilePoint Development Considerations..... 42
Windows and Pages that Refresh Automatically.....43
Task List Control Placement.....43
Database Connection Sizes.....43

Preface

Disclaimer of Warranty

AgilePoint, Inc. makes no representations or warranties, either express or implied, by or with respect to anything in this document, and shall not be liable for any implied warranties of merchantability or fitness for a particular purpose or for any indirect, special or consequential damages.

Copyright

Trademarks

AgilePoint, Inc. and AgilePoint's products are trademarks of AgilePoint Inc. References to other companies and their products use trademarks owned by the respective companies and are for reference purpose only.

Government Rights Legend

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the applicable license agreement and as provided in DFARS 227.7202-1(a) and 227.7202-3(a) (1995), DFARS 252.227-7013(c)(1)(ii) (Oct 1988), FAR 12.212(a) (1995), FAR 52.227-19, or FAR 52.227-14, as applicable.

Virus-free software policy

AgilePoint recognizes that viruses are a significant security consideration for our customers. To date, we have had no report of AgilePoint BPMS carries any virus. AgilePoint takes the following measures to ensure our software is free of viruses upon delivery:

- AgilePoint is built on top of Microsoft .NET framework. The pre-compiled executable is a.NET Common Language Runtime (CLR) application, not a native machine binary. As far as is known at this time, there are no viruses that infect .NET CLR executables.
- The virtual environment for the product packaging process in is fully isolated and protected, and anti-virus software is installed and running during packaging.
- The deliverable package is scanned by anti-virus software before upload to our customer download site.

Document Revision Numbers

AgilePoint documentation uses the revision number format **r*X.Y.Z***. The letters and numbers in this revision number can be interpreted as follows:

- **r** - Indicates "revision." This helps to differentiate the document *version* numbers, which start with **v**.
- ***X*** - The major version number for AgilePoint BPMS to which this document refers. For example, AgilePoint releases 5.0, 5.0 SP1, and 5.5 would all have an ***X*** value of **5**.
- ***Y*** - The major document revision number. This number typically changes only when either there is a new AgilePoint release, or there are major changes to the document.
- ***Z*** - The minor document revision number. This number is incremented each time the document is republished.

AgilePoint Documentation in PDF and HTML

AgilePoint documentation is provided in both print-friendly (PDF) and web-based (HTML) formats.

Advantages of HTML Documentation

- HTML is the **primary delivery format** for AgilePoint documentation.
- Unified, global **search** across all documentation. PDF documents allow you to search only within the context of a given PDF file.
- **All hyperlinks supported**. Links in PDFs are only supported in certain contexts.
- "One-stop shopping" for all information related to AgilePoint BPMS.
- The HTML documentation is updated more frequently than the PDF documentation. Web-based documentation is updated periodically between AgilePoint releases to address errors and omissions, but the PDF documentation is updated only at the time of a software release.

Advantages of PDF Documentation

PDFs can be more easily **printed**, **archived**, and **transferred** (such as by FTP or email) than HTML documentation.

For more information, see Downloading Files and Sharing Links from the Documentation Library in the Documentation Library.

Opening the Documentation Library

To open the AgilePoint Documentation Library, do the following.

Prerequisites

You must have a valid account on the AgilePoint Support Portal.

Instructions

1. Log on to the AgilePoint Support Portal.
2. Click **Documentation**.
3. On the **Documentation** page, click the documentation library for your AgilePoint release.
    - For AgilePoint BPMS v5.0 SP1 and higher, the web-based documentation library opens in a new tab or window in your web browser.

- For releases prior to v5.0 SP1, a download starts for a Zip file with the PDF documentation for your release.

Finding Information in the Documentation Library

The information in this topic will help you to locate information in the AgilePoint Documentation Library.

Using the Table of Contents

The table of contents in the AgilePoint Documentation Library is divided by content areas. For example, the Installation section includes all the information you need to install AgilePoint BPMS. The AgilePoint API section includes information about the AgilePoint APIs.

You can use the Table of Contents to explore the AgilePoint documentation content and find the information you want.

Searching

The web-based documentation includes a centralized search for all documentation content. To search for information:

1. In the AgilePoint Documentation Library, click the **Search** tab. In the Search box, enter **1 search team**, and click **Search**.

   The search results display in alphabetical order by topic title.

   It is important to understand that the third-party software AgilePoint uses to generate web-based documentation allows only 1 search term. More than 1 search term will cause the search to fail.

   AgilePoint recommends using a relatively unique search term to find the information you need. For example, entering a common term, such as "process," will return a high percentage of the total documentation topics in the search results.

2. Browse the list of topic titles to find the information you want.

Printing

The PDF documentation is provided mainly for the purpose of printing and archiving. To print a set of information:

1. Navigate to the main page of the Documentation Library from which you want to print.
2. In the list of documents, click the document name in the **PDF** column.
3. From your PDF reader software, print the portion of the document you want.

Downloading Files and Sharing Links from the Documentation Library

You can download and share files AgilePoint's documentation library as you would in any other web page. Note that if you send links to recipients, they must have a Support Portal login to view the file.

These procedures are common examples based on Internet Explorer with the Adobe Reader plug-in. Exact procedures may vary depending on your web browser, PDF viewer, and email client configuration.

Share a Link to an HTML Topic

1. Navigate to the topic you want to share.
2. Copy the URL in the Location box in your web browser.
3. Paste the URL in an email, IM client, etc.

Share a Link to a PDF Document

1. In Internet Explorer, navigate to the Documentation Library home page.
2. In the **PDF** column, right-click the name of the PDF file you want to share.
3. In the quick menu, click **Copy shortcut**.
4. Paste the URL in an email, IM client, etc.

Save a Copy of a PDF Document

1. In Internet Explorer, open the Documentation Library home page.
2. In the **PDF** column, click the name of the PDF file you want to share.
3. In the Adobe Reader plug-in, click **Save** button.

Contacting AgilePoint Sales

AgilePoint is a leading Business Process Management System (BPMS) provider created by a team of driven people who strive to incorporate the principles of relentless innovation for the benefit of our customers. Our mission is to help companies of any size attain and sustain operational success through process excellence.

Headquarters: AgilePoint Corporation 1916C Old Middlefield Way Mountain View, CA 94043, USA

Tel: (650) 968 - 6789

Fax: (650) 968 - 6785

Email: info@agilepoint.com

Web site: www.agilepoint.com

International: For AgilePoint EMEA and AgilePoint Asia Pacific, please call the AgilePoint Corporate Office for contact information.

Contacting Customer Support

To contact AgilePoint Support, please submit a ticket on the AgilePoint Support Portal: http://support.agilepoint.com/SupportPortal/

If you do not have a Support Portal account, you can send an email to request one: support@agilepoint.com

Scalability

To begin, scalability refers to the ability of an application to continue to meet its performance objectives with increased load. Typical performance objectives include application response time and throughput. AgilePoint Server provides both Scale Up and Scale Out flexibility options to handle load.

Scale Up

Scale Up is typically achieved by upgrading with better hardware (faster CPU, memory, disk, network adapters, etc.). Scale Up techniques can be employed on AgilePoint servers as it supports multi-threading, highly concurrent processing, and also 64-bit operating systems by bumping up hardware capacity.

Scale Out

Scale Out is typically achieved by spreading load across multiple servers via dedicated servers and load balancing cluster deployments. In addition to dedicated server deployments which allows for an increase in the total amount of hardware resources available to the system, the AgilePoint BPMS suite also supports clustering that supports the ability to dynamically allocate load across the cluster according to the systems needs.

Capacity of the Machines in your Environment

AgilePoint offers a baseline recommendation based on minimum recommended hardware. You may need fewer machines if each machine in your environment has a higher capacity. The main hardware considerations for an AgilePoint are covered in this section.

CPU Speed

Increasing the server's CPU speed will increase the amount of work that can be performed by the server within a given time frame. Therefore, faster CPUs will almost always improve performance. However, one notable exception is that a faster CPU will not (necessarily) resolve performance issues that stem from performance bottlenecks.

Number of CPUs

Since AgilePoint Server is multi-threaded and is designed to support asynchronous processing, AgilePoint Server can take full advantage of the benefits of multiple server CPUs. Multiple CPUs will generally improve performance for any real-world AgilePoint system because it offers the greatest improvement for systems that will have concurrent usage by multiple users. However, it may not have as great an affect on the performance of an AgilePoint system that is only used by one or a few users at a time (such as a development or QA system). As with CPU speed, multiple CPUs may not (necessarily) resolve performance issues that stem from performance bottlenecks, although it may prevent a bottleneck from affecting some of the system's users.

CPU Sizing

CPU Sizing is based on the number of transactions per CPU per second under optimal and regular conditions.

The typical formula for computing CPU sizes is as follows:

Number of transactions = number of concurrent requests * (optimal/regular response times) 0.80 * (Speed of CPU in Mega Cycles) * No. of CPUs

= Mega Cycles per Transaction * No. of Transactions

The reason we use the 0.80 factor is because the threshold for CPU utilization before it is considered a bottleneck is 80%. If you want to take into account future growth you should use a smaller factor.

System Memory (RAM)

Unlike CPU speed, where slower CPUs will just require more time for the server to finish performing its work, the server's physical and available memory impose an upper limit to the size and quantity of the data and work that the server can be working on simultaneously. Therefore, for mission critical systems, maximizing the physical memory is always best.

Memory Utilization

Memory requirements for AgilePoint Server depends on several factors such as:

- Number of active process instances
- Typical amount of process instance size varies by amount of custom data carried along process
- Custom application data at each step
- Number of concurrent requests

Note: It is a best practice for memory utilization to have a dedicated Application Pool in IIS for AgilePoint Server.

Hard Disk Performance

Since data will be read from and saved to the server's hard disk, the hard disk's seek time, read response, RPMs, etc. will affect the performance of AgilePoint systems. Therefore, the faster your server's hard disk performs, the better your system's performance will be. However, hard disk performance generally has the greatest affect on database servers. For other servers (or for hard disks that don't contain a database), hard disk performance is much less critical and unlikely to affect performance significantly, unless the server starts using virtual memory (see above), in which case it may become very critical.

Network Bandwidth

Network Bandwidth is determined by amount of data pulled/pushed per request. This is usually determined by the number of active requests at any particular instance. Please note, you need to consider bandwidth between client applications and the AgilePoint Engine as well as bandwidth available between the database and AgilePoint Server.

Rule of thumb for network bandwidth:

There should be at least 1 GB bandwidth between AgilePoint and Database Server and 10 – 100 MB is sufficient between client applications and AgilePoint Server.

Network Adapters

Usually network bandwidth is more of an issue than network adapters. However, the server's network adapters should be sufficiently fast and reliable to allow the server to make maximum use of the full bandwidth provided by the networks the server is connected to. Anything slower (or an unreliable adapter) could represent a potential performance bottleneck.

Virtual Environments

Using a virtual environment for AgilePoint is much cheaper than physical servers, although performance tends to be slower.

Because of the reduced performance, it is not possible to run load tests on a virtual environment, but if load testing is not required, AgilePoint recommends using a virtual environment. This is an especially effective approach for development environments.

Even if you use a virtual environment, physical database servers are recommended.

Development in Virtual Environments

In particular, virtualization is advantageous for development environments because it enables isolation (sandboxing) between development environments to prevent them from impacting one another. Using virtualization technology also speeds up the process of environment configuration. It is a best practice to keep copies of each version of development environment for future enhancement.

Support for Virtual Environments

AgilePoint is committed to fully supporting AgilePoint BPMS running on virtualization technologies. AgilePoint recommends installing AgilePoint on a physical server machine, but AgilePoint can be supported in a virtual machine (including NLB) given the virtual machine can support Windows Server 2003 or 2008, and the .NET Framework 3.5. AgilePoint recommends Windows Server® 2008 Hyper-V™, but other Microsoft and non-Microsoft virtualization products are also supported as discussed here. In addition to Windows Server® 2008 Hyper-V™, AgilePoint can also be deployed using Microsoft Virtual Server and Microsoft Virtual PC virtualization technologies. Only virtualization products that have specifically passed Microsoft's requirements for virtualization support are also officially supported for running AgilePoint. See the following Microsoft Web site that provides a list of non-Microsoft virtualization products that have passed the requirements for Windows Server 2003 and 2008:

http://www.windowsservercatalog.com/results.aspx?&bCatID=1521&cpID=0&avc=0&ava=0&avq=0&OR=1&PGS=25&ready=0

AgilePoint recommends the following configurations to ensure optimal performance of the AgilePoint BPMS in a virtual environment:

Host OS Architecture	X64
Host Machine Processors	4
Memory	4 GB allocated to each virtual machine
Bandwidth	1 Gb Network connection to the host machine
Virtual Memory	Memory allocation cannot always be guaranteed (i.e. if 4 GB is set for the virtual machine, it does not mean that the virtual machine will be using the entire 4 GB of memory at all times). The memory is shared amongst all the virtual machines.

AgilePoint Database	The AgilePoint Database should be on a physical machine.

Dedicated Servers

A dedicated server, multiple machine scenario is generally best for the following profiles:

- High availability, small capacity enterprise or standard production environments
- High availability, medium capacity enterprise
- High availability, large capacity enterprise

In most cases (especially production environments) where there will be many concurrent users and/or large amounts of data, a single all-in-one server may not be capable of adequately performing all of the functions required with a satisfactory level of performance (or in some cases, a single server may not be capable of doing everything period). In this case, you will probably need to move/split some or all of the system's functionality (e.g. database, workflow, SharePoint, Active Directory, etc.) onto separate, dedicated servers. This will allow you to increase the total amount of hardware resources available to the system, but at the cost of transferring additional data over the network between the various servers.

Non-Dedicated Server

A non-dedicated, single machine scenario is generally best for small capacity enterprise deployments, or standard development environments. This scenario completely avoids any potential bottlenecks or performance issues related to the interaction and transfer of data between the servers. However, this setup forces all of the software applications that reside on the server to share the same hardware resources (e.g. CPUs, RAM, etc.). If network bandwidth/performance is a major issue, and if your system's usage patterns and data metrics can be reasonably supported by a single all-in-one server, then that may be the best option for ensuring optimal performance of your entire AgilePoint system.

Clustered Servers

Server clustering provides a platform for scalability and high availability in which multiple AgilePoint Servers can be setup to distribute load and/or eliminate a single point of failure. Server clustering ensures high availability and data integrity by providing failover functionality across two or more connected server machines. You can use server clustering in a single AgilePoint Server deployment, or in a NLB environment.

- For single AgilePoint Server deployment scenarios, if the active AgilePoint Server machine fails, the cluster service enables failover to a redundant AgilePoint Server machine.
- For a NLB cluster AgilePoint Server deployment, the cluster service maintains the availability of the AgilePoint CSM application.

Clustering Terminology

It is important to understand the role of the NLB, Cluster Service, and the AgilePoint CSM as it pertains to an AgilePoint deployment.

- **NLB Cluster** - Performs the load-balancing of IP traffic.
- **AgilePoint CSM** - Ensures all events of a single process instance are processed on the same AgilePoint Server machine.
- **Cluster Service** - Provides failover capability to maximize high availability of the servers.

NLB Cluster

NLB load-balances incoming client IP requests across multiple AgilePoint Servers in the cluster. The AgilePoint Servers in the cluster then concurrently process the different client requests.

NLB is setup so that each AgilePoint Server in the load balanced group is configured with the same Virtual IP address. Whenever a request is made on this Virtual IP, a network driver on each of these machines intercepts the request for the Virtual IP address and re-routes the request based on the load to one of the machines in the NLB server group.

To implement NLB for AgilePoint Server, an NLB software or hardware device is required. Microsoft offers a low cost, Network Load Balancing Manager software component as part of Windows Server. While Windows Server provides software network load balancing, an alternative high performance load balancing hardware device may be better suited for production systems.

Scalability

The NLB cluster can be scaled out by adding additional servers to ensure performance as load increases, allowing the ability to start small and grow as needed.

To scale in an existing NLB environment, additional CPUs and systems can be added to the NLB cluster in a simple plug-in fashion.

High Availability

A high availability cluster includes two or more machines that are configured so that, if one AgilePoint Server machine is brought down in the event of planned downtime due to maintenance or unplanned downtime due to hardware, OS, or application failures, the CSM automatically redirects incoming requests to the other AgilePoint Servers (nodes) in the cluster, maintaining service and application availability without interruption.

While processing server requests, each AgilePoint Server in the cluster sends out heartbeats to the other AgilePoint Servers in the cluster. If an AgilePoint Server fails, it stops sending heartbeats, then after a default time period of five seconds, the failed AgilePoint Server is removed from the cluster and new requests are mapped to the remaining AgilePoint Servers in the cluster. This is all handled by the NLB device.

AgilePoint CSM

AgilePoint Clustering Server Manager (CSM) is a feature that enables AgilePoint Server to operate effectively in a network load balanced (NLB) or failover environment.

Below are some important notes about AgilePoint CSM:

- The AgilePoint CSM enables AgilePoint BPMS to work with NLB, but it is not NLB software.
- NLB environments may require specific hardware, software, or operating system options that are not covered in the AgilePoint BPMS System Requirements. For more information, see your network system administrator.

How the AgilePoint CSM Works

The AgilePoint CSM runs as a Windows Service and works as an event dispatcher using .NET Remoting. (In AgilePoing v5.0 R2 SP1 and higher, this is part of the Windows service for AgilePoint Server.) It records information from the AgilePoint Servers in the NLB to track which process instances are running on each of the AgilePoint Server machines. When a process instance is created the first event will be load balanced by the NLB, after that, the AgilePoint CSM is involved to dispatch events to each of the AgilePoint Servers in the cluster, ensuring that all events associated with the same process instance lifecycle are processed on the same AgilePoint Server machine. As the AgilePoint Server machines monitor the DB, when a new event is triggered, the AgilePoint Server machine picks up the event and posts a call to the AgilePoint CSM to determine whether the event is currently being handled by any of the AgilePoint Servers in the NLB based on the associated Process Instance ID. If the event is associated with a Process Instance ID that is already running on a different server, The AgilePoint CSM will re-route the event to be processed on the AgilePoint Server that is currently dedicated to processing events associated with that process instance.

Flow of Events in a NLB Deployment

The following information explains the basic flow of events in a NLB deployment when an AgilePoint process instance is created. Assume that only two AgilePoint Servers are in the NLB cluster (e.g. AgilePoint Server A and B).

1. When a new AgilePoint process instance is created, the NLB manager load balances this request and distributes it to AgilePoint Server A in the NLB cluster.
2. AgilePoint Server A will do the following:

- Create the records in the AgilePoint database.
- Sends a message to the AgilePoint CSM with the Process Instance ID to dedicate AgilePoint Server A to process all activities associated with this process instance.

3. As both AgilePoint Servers monitor the DB for new events, only the AgilePoint Server dedicated to a process instance will process the event associated with a current running process instance. In other words, when a new event becomes available in the database, AgilePoint Server B will pick up the event and do a check with the AgilePoint CSM. The AgilePoint CSM replies that AgilePoint Server A is dedicated to processing events for this process instance and blocks AgilePoint Server B from processing the event. AgilePoint Server A will process the event.

General Performance Considerations

Below are three important factors when considering the performance of an AgilePoint BPM system:

- The system's usage patterns
- The size and/or amount of data the system interacts with
- Performance bottlenecks

System Usage

Three main factors go into determining system usage for AgilePoint:

- The number of concurrent users
- The number of process steps to be completed per day
- External system load

Note that again, the guidelines provided here apply in most cases. If you have questions regarding your specific configuration needs, contact an AgilePoint representative.

Concurrent Users

The number of concurrent users refers to both process participants and users who are not part of the process but access the system to gather information, such as process status. This number affects the number of front-end application machines you will need.

Process Steps Per Day

To determine your system capacity, you must determine your process steps per day.

Maximum Process Steps per Day for AgilePoint

Use the following formula to estimate how many steps can be completed per day:

Total steps that can be completed in 24 hours = [Thread pool size]*24 (hrs)*60 (m)*60(s)*1,000(ms) /[Average event processing time]*3

Assume a typical server with 2 CPUs (Xeon 3.6 GHz), dual-core and 4GB memory and connects similar configuration of database server. The average AgilePoint event processing speed is 321(ms)/per event (3 events for one step). Assume a thread pool capacity of 30.

Total steps that can be completed in 24 hours = 30 *24 *60 *60 *1,000/321*3 = 89,719 steps/day

Estimating Your Process Steps Per Day

The number of process steps you require per day can be determined using a worksheet and calculations provided by AgilePoint. In order to set up your worksheet, we require the following information:

- The number of years you would like to project
- The number of processes you foresee creating during that time span

Once we have calculated your estimated process load per day, we can compare it to the following calculations.

Once your process steps per day are calculated, we compare it to the maximum of 89,719. If your requirements are less than this number, the minimum hardware configuration is sufficient.

External System Load

External system load refers to any external systems that will access the AgilePoint system. External system load is a variable that can impact your performance, but it is difficult to calculate precisely what the impact will be.

AgilePoint recommends using either an educated guess based on your knowledge of the external systems to determine the impact, or to run test metrics to more precisely determine the load. If you have questions regarding your specific configuration needs, contact an AgilePoint representative.

Data Metrics

The quantity and size of the data the system interacts with also has a great impact on the system's overall performance. As would be intuitive, systems that will have to deal with larger quantities of discrete data items (e.g. documents, database records, etc.) must do more actual work than systems that deal with smaller quantities. Also, systems that deal with data items of larger sizes (e.g. individual documents with sizes measured in multiple megabytes) must often perform more actual work than systems that deal in a similar quantity of smaller-sized data items.

This means that systems that interact with larger quantities and/or sizes of data will usually need to have more powerful hardware and/or different deployment and configuration (versus lower-load systems) to ensure optimal performance.

Performance Bottlenecks

As opposed to the factors above, which deal with the actual amount of work the system must accomplish, a performance bottleneck represents a situation where (for whatever reason) some part of the system's resources are in use, but those resources are prevented from performing actual work. For example, this may occur if the AgilePoint system must interact with external systems, and is forced to wait for the external system to return requested data before the AgilePoint system will be able to continue performing more work. While this sort of scenario may not adversely affect performance at all (if properly planned for and implemented), it may tie up critical system resources (e.g. memory and CPU cycles) while waiting, that could instead be used to perform other work.

The best way to resolve bottleneck issues is to identify possible bottlenecks ahead of time, and to implement your processes in a way that will not tie up the system's resources unnecessarily when a potential bottleneck scenario occurs.

Network Bandwidth

Note: Network bandwidth is usually only a factor when dedicated servers are used, since all-in-one type servers don't need to access the network much while performing their work.

One of the primary causes of performance bottlenecks in AgilePoint systems is insufficient network bandwidth and/or performance. Determining whether your network bandwidth is sufficient depends greatly upon the system's usage patterns and data metrics. In fact, depending on an AgilePoint system's usage patterns and data metrics, improving network bandwidth and performance can sometimes improve performance more than using faster hardware. For example, for system's that handle large amounts of data (in either quantity and/or size), the bandwidth and throughput between the workflow server and the workflow database server can be critical to achieving optimal performance. Similarly, for AgilePoint systems that are integrated with SharePoint, the bandwidth and throughput between the workflow server and the SharePoint server (and also between the SharePoint server and the SharePoint server's database server) is often very important.

However, increasing network bandwidth for improving network performance is sometimes not an option, because network throughput is usually a factor of the network's infrastructure. Therefore, when network issues are causing significant performance issues, it is sometimes effective to combine dedicated servers into a single non-dedicated server. Merging dedicated servers into one completely eliminates any network performance issues resulting from the traffic between the merged servers, since all such traffic now remains entirely on the merged server. However, merging dedicated servers may also cause you to lose the advantages that having dedicated servers provided (see below), so this option should be considered very carefully before being taken.

Another possible improvement is to physically relocate some or all of the system's servers, as this may also improve network throughput between the servers without requiring major infrastructure changes.

AgilePoint Server Performance Characteristics

AgilePoint Server is implemented by employing Microsoft .NET technology. AgilePoint leverages data caching technologies across all tiers, supports process hydration/dehydration (process swapping) which enables better memory management, and AgilePoint metadata store employs several database techniques to optimally configure all these techniques for the best performance. The AgilePoint Server architecture supports distributing entities such as AgileParts, Sub Processes, or a process activity of a particular process to be processed by its dedicated server (Server Cluster), effectively accomplishing the application context based load balancing concept which provides flexible performance tuning options.

Event Driven Architecture

AgilePoint uses an event driven architecture. Events are executed, the resulting state is recorded, and the objects are released to free server memory and CPU cycles. Another AgilePoint design feature is that execution of events is not limited to process based threading. If a process has several activities assigned to different participants, the events of the activities can execute in different threads.

AgilePoint maintains the state of all workflow objects in the database and only hydrates them (creates the appropriate workflow object on the AgilePoint server) when an event is generated from a call to the engine or a timed event is created by the AgilePoint Timer service. The AgilePoint Timer service monitors the database and creates events based on date and time of workflow objects. An example of a timed event is an activity that has not been completed by its due date. An event will be fired to change its status from "Assigned" to "Overdue". Concurrently, if a process instance is in idle state (e.g. a task is assigned to a user and is waiting for the user's response, a call to an asynchronous Web service, or waiting for a sub process to complete) it does NOT consume any memory or CPU/thread. As a result, more often than not, the process is in idle state for human oriented workflow.

Time-Sharing Model

AgilePoint Server employs a time-sharing model where the maximum number of threads is configurable based on hardware specifications and characteristics of the process (i.e. whether there are more human oriented or system oriented activities). For more information, see Configuring the AgilePoint Server Thread Pool Size in the Documentation Library.

For example, if maximum threads = 100, and there are 50,000 running processes, 10,000 are active (not idle), only a maximum of 100 threads will be created and shared by 10,000 process instances.

When process instances become in idle state, the workflow engine starts swapping process instances from memory. Or, when total memory (.NET) reaches a certain point, the workflow engine swaps some of process instances out of memory based on the last processing timestamp. When the engine has nothing to do, it will release threads and leave 5 threads at a minimum.

Interruption Handling

If an IIS restart or server reboot is necessary, the state of the workflow objects is maintained in the AgilePoint database. Any workflow objects that were active when the AgilePoint engine was interrupted, will be restarted once the AgilePoint service restarts. Furthermore, AgilePoint does not execute workflows using process based assemblies for execution. It uses an event driven architecture that is designed to optimize server performance

by recording the state changes associated with an event in the database and releasing server resources. The AgilePoint server manages the threads to provide the best possible server performance characteristics.

Predicting AgilePoint Server Resource Usage

Although the AgilePoint Server application normally runs constantly and requires a certain amount of server resources, it is optimized so that server resources are not normally consumed for data and processes that are not currently being actively worked on, interacted with, or updated. This allows AgilePoint Server to potentially interact with millions of separate process instances over time using limited CPU and RAM at any given time. However, when trying to estimate usage patterns and predict server load and resource usage, the following points should be considered.

AgilePoint Server's core engine is normally active and running at all times, even if no users or external systems are currently interacting with AgilePoint. This engine is responsible for various functionality, such as handling requests to the Workflow API and for scheduling and coordinating time-dependent events (e.g. reminder emails). The core engine is optimized to consume as little of the server's resources as possible, but it will require some portion of the server's CPU cycles and system memory.

However, in addition to the minimum server resources used by the core engine itself, additional server resources will be required and used when specific work or processing occurs. Examples of things that will initiate additional processing and (temporarily) increase the server's load include:

- Processing of requests resulting from direct interaction with the AgilePoint system by users.

  For more information, see Memory Utilization in the Documentation Library.
- Processing of requests resulting from direct interaction with the AgilePoint system by external systems
- Processing (if necessary) of the escalation scenario defined by the process model for a task or activity that has "timed out" (i.e. become overdue)
- Processing (as needed) of time-dependent functionality (e.g. reaching the end of the configured time interval for a Delay AgileShape)
- Processing (as needed) of scheduled and/or recurring operations, such as Active Directory synchronization, or archiving of old workflow data
- The escalation scenario defined by the process model for a task or activity that has "timed out" (i.e. become overdue)

All of the above represent a request (whether direct or indirect) for AgilePoint to perform additional work and processing, and the amount of server resources required and used to perform that processing will depend on the exact nature of the processing that needs to be performed.

Performance Calculations

AgilePoint performance levels can be calculated using the information detailed below.

Calculating Average Event Processing Time

AgilePoint performance levels can be calculated using a formula as detailed below. You must first calculate the average event processing time. The average event processing time can be calculated as the sum of the time span for all internal and external events that are processed by the workflow engine divided by the total # of events. Internal events for example could be the execution of AgileParts, conditional activities, etc. External events are processed by external entities such as the Web Services, API, and Remoting API.

All events = internal events + external events.

Average event processing time = time span for all events/total # of events.

External events are recorded in the AgilePoint Database table WF_EVENTS. You can evaluate the columns SENT_DATE and END_DATE to calculate the time span.

Time span = END_DATE - SENT_DATE.

How a Faster Workflow Engine Can Increase Capacity

The following data provides some insight into how a faster workflow engine can increase capacity. The following scenario involves a process running an AgilePart calling a Web Service through the Internet.

On an average server:

Average event processing time is 551ms/event. Thread pool size = 10.

From activity to activity requires 3 events, hence 1 activity = 3 events.

Assuming one process instance contains 30 activities, and the tasks under each activity are completed immediately once assigned (no waiting time).

Time needed to complete 1 process using 1 thread:

= 551 ms/event x 3 event/activity x 30 activity/process

= 551 x 3 x 30 (ms/process)

Number of processes that can be completed in 1 day using 10 threads:

$$= \frac{\text{Total time available in 24 hours (ms) x Thread pool size}}{\text{Time needed for 1 process (ms/process)}}$$

$$= \frac{\text{24 hours/day x 60 min/hr x 60 sec/min x 1000 ms/sec x 10 threads}}{\text{551 x 3 x 30 (ms/process)}}$$

= 17, 422 process/day

Same process on a faster server (Xeon 3.6 * 2):

The average event processing speed is 321(ms)/per event.

Thread pool size = 50.

Total processes that can be completed in 1 day

= 50*24*60*60*1,000 / 321*3*30

= 149,533 process/day

The formula is:

Total processes that can be completed in 24 hours:

= [Thread pool size]*24*60*60*1,000/[Average event processing time]*3* [Number of activities per process]

Or

Average event processing time

= [Thread pool size]*24*60*60*1,000/[Total processes that can be completed in 24 hours]*3* [Number of activities per process]

Scenario to Consider

Consider the following scenario:

Assuming:

- Average event processing time using 1 thread = 551ms/event
- 3 events are fired per activity
- 1 process contains 30 activities
- The tasks under each activity are completed immediately once the tasks are assigned (no waiting time).

Scenario 1

What is the total time to complete 4,000 process instances?

Total time to complete 4,000 processes using 10 threads:

= 551 ms/event x 3 events/activity x 30 activities/process x 4000 processes
10 threads

=19,836,000 ms

=19,836 s

Total time to complete 4,000 processes using 20 threads:

= 551 ms/event x 3 events/activity x 30 activities/process x 4000 processes
20 threads

=9,918,000 ms

=9,918 s

Total time to complete 4,000 processes using 40 threads:

= 551 ms/event x 3 events/activity x 30 activities/process x 4000 processes
40 threads

=4,959,000 ms

=4,959s

Scenario 2

With the hardware combination discussed in Section 7.2, how may processes can be completed in 1 day?

Number of processes that can be completed in 1 day using 10 threads:

= Total time available in 24 hours (ms) x Thread pool size

Time needed for 1 process (ms/process)

=17, 422 process/day

Number of processes that can be completed in 1 day using 20 threads:

= 24 hrs/day x 60 min/hr x 60 sec/min x 1000 ms/sec x 20 threads

551 x 3 x 30 ms/process

=34, 846 process/day

Number of processes that can be completed in 1 day using 40 threads:

= 24 hrs/day x 60 min/hr x 60 sec/min x 1000 ms/sec x 40 threads

551 x 3 x 30 ms/process

=69, 691 process/day

Scenario 3

With this kind of hardware combination, how may processes can be completed in 1 week and 1 month?

Using the results from Scenario 2, and assuming 1 month = 30 days.

Thread Pool Size	Total processes that can be completed in 1 week	Total processes that can be completed in 1 month
10	121, 960 processes/week	522, 686 processes/month
20	243,920 processes/week	1,045,372 processes/month
40	487,840 processes/week	2,090,744 processes/month

Performance Tracing Utility

The Performance Tracing utility allows you to report on performance of custom components such as custom AgileParts, AgileWorks, AgileConnectors, AgileStubs, or Web Services. The performance criteria is based on milliseconds, for example you can write a query to return the most time consuming AgileParts. To use this utility, you will need a database that includes a table called WF_PERF_TRACES (it is not recommended to add this table to the AgilePoint database). You will also need to add a PerfTrace.config XML file to the root level folder where AgilePoint Server is installed (e.g. c:\inetpub\wwwroot\AgilePointServer\). The location of AgilePoint Server can be determined from the registry.

To use the utility to perform performance tracing:

1. Create a new table called **WF_PERF_TRACES** in a database as below. It is not recommended to use the AgilePoint database for this table.

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [WF_PERF_TRACES](
[SOURCE] [varchar](1024) NULL,
[CATEGORY] [int] NULL,
[OBJECT_ID] [varchar](256) NULL,
[TIME_STARTED] [datetime] NULL,
[TIME_SPAN] [int] NULL
) ON [PRIMARY]
GO
SET ANSI_PADDING OFF
```

   A sample query may return results as shown below:

	SOURCE	CATEGORY	OBJECT_ID	TIME_STARTED	TIME_SPAN
1	#AgilePart:Ascentn.AgileP...	0	b81fad11691e4ce7a6a92...	2007-08-03 15:56:47.350	171736
2	#AgilePart:Ascentn.AgileP...	0	77fb50b240dc422ebf6e9e...	2007-08-03 16:08:01.440	83109
3	#AgilePart:Ascentn.AgileP...	0	0e5a6fc77c6349f3a86564...	2007-08-03 19:22:50.990	54468
4	#AgilePart:Ascentn.AgileP...	0	5342b4a43d764c4a8b3ca...	2007-08-01 17:25:30.700	54338
5	#AgilePart:Ascentn.AgileP...	0	e3f8d99521bd481d8e6c3...	2007-08-01 19:26:36.017	47184
6	#AgilePart:LabOneAgilePar...	0	3621C4E80B004EBA97E0...	2007-07-20 03:38:29.173	35541
7	#AgilePart:LabOneAgilePar...	0	C19819A8BC6A4FAEA0A...	2007-07-20 03:38:29.253	35460
8	GetEvent	3	T42Z\Administrator	2007-08-05 21:22:11.120	31124
9	#AgilePart:Ascentn.AgileP...	0	e0e9a3590dd94b1984be3...	2007-08-03 17:16:47.443	20229
10	#AgilePart:Ascentn.AgileP...	0	b81fad11691e4ce7a6a92...	2007-08-03 16:00:28.307	16473

2. In a text editor, create a new file with the following content:, and save it with the name **PerfTrace.config**.

```
<?xml version="1.0" encoding="utf-8"?>
<PerformanceTraceSetting xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <Database>
      <Vendor>MSSQLDatabase</Vendor>
      <Connstr>server=localhost;database=lock;trusted_Connection=yes</Connstr>
```

```
  </Database>
  <Filter>
     <Category>TrackingEvent</Category>
     <Category>WebService</Category>
     <Category>ProcessEventHandler</Category>
     <Category>AgilePart</Category>
  </Filter>
  <MinimumTimeSpan>10</MinimumTimeSpan>
</PerformanceTraceSetting>
```

3. Save the file to the following path and file name:

```
[AgilePoint Server installation folder]\PerfTrace.config
```

4. Configure **PerfTrace.config** as follows:

   a. Within the **Database** section, enter the database vendor and connection string for your performance tracking database.

   b. The **Filter** section allows you to add the categories for which you want to report on. The categories that are available include:

      - **AgilePart** – Reports on all AgilePart data.
      - **ProcessEventHandler** – Reports on AgileWork and AgileStub data.
      - **TrackingEvent** – Reports on AgileConnector data.
      - **WebService** – Reports on Web Service data.

   When doing the reporting, a few of the filtering options that are available include:

- **Exclude Categories** - To exclude one or more of the categories, remove the category from the PerfTrace.config file. For example, remove the <Category>WebService</Category> to tell AgilePoint Server to exclude Web Service tracing data from being recorded to the database.
- When writing your SQL, each category has been assigned a numerical identifier for filtering. This allows you to report only on a certain category. Each category is identified as follows:
  - **AgilePart** – 0.
  - **ProcessEventHandler** – 1.
  - **TrackingEvent** – 2.
  - **WebService** – 3.

A couple of sample query statements are shown below:

```
select * from WF_PERF_TRACES where source like
'%Sample%' order by time_span desc
```

```
select * from WF_PERF_TRACES
where  category = 0 and source like '%Move%'
order by time_span desc
```

A sample query may return results similar to below:

	SOURCE	CATEGORY	OBJECT_ID	TIME_STARTED	TIME_SPAN
1	#AgilePart:Ascentn.AgileP...	0	b81fad11691e4ce7a6a92...	2007-08-03 15:56:47.350	171736
2	#AgilePart:Ascentn.AgileP...	0	77fb50b240dc422ebf6e9e...	2007-08-03 16:08:01.440	83109
3	#AgilePart:Ascentn.AgileP...	0	0e5a6fc77c6349f3a86564...	2007-08-03 19:22:50.990	54468
4	#AgilePart:Ascentn.AgileP...	0	5342b4a43d764c4a8b3ca...	2007-08-01 17:25:30.700	54338
5	#AgilePart:Ascentn.AgileP...	0	e3f8d99521bd481d8e6c3...	2007-08-01 19:26:36.017	47184
6	#AgilePart:LabOneAgilePar...	0	3621C4E80B004EBA97E0...	2007-07-20 03:38:29.173	35541
7	#AgilePart:LabOneAgilePar...	0	C19819A8BC6A4FAEA0A...	2007-07-20 03:38:29.253	35460
8	GetEvent	3	T42Z\Administrator	2007-08-05 21:22:11.120	31124
9	#AgilePart:Ascentn.AgileP...	0	e0e9a3590dd94b1984be3...	2007-08-03 17:16:47.443	20229
10	#AgilePart:Ascentn.AgileP...	0	b81fad11691e4ce7a6a92...	2007-08-03 16:00:28.307	16473

Note: It is not necessary to restart IIS upon making changes to the configuration file. AgilePoint Server will begin recording this tracing data to the database within 10-30 seconds. You can delete this file or rename it to stop this service.

Performance Testing and Benchmarking

AgilePoint's heritage is in enterprise strength workflow management solutions and AgilePoint's engine architecture reflects that with superior resource and event optimization to achieve highly efficient scaling on a single server or across a distributed environment.

AgilePoint's technical sophistication was born of the founders' experience in vendor competitions at one of top three financial institutions where test criteria and load guidelines were extremely stringent and demanding. This global enterprise class experience base was the foundation for AgilePoint's design criteria and guidelines.

Stress Testing

AgilePoint's enterprise class architecture was designed to scale up and scale out in high transaction volume enterprise environments. While AgilePoint can be implemented in a variety of configurations to take advantage of clustering, component server and distributed application execution capabilities, third party stress tests were performed with a single AgilePoint server configuration to provide a baseline for performance evaluation.

Scenario #1 – Single AgilePoint Server Concurrent Process Scalability

Result at Peak: Initiated 125,000 concurrent process instances with 10 activity steps without reaching a capacity peak.

Hardware configuration: Pentium machine with 2.0 GHz, 2GB RAM, on 2 Processors

Operation System: Windows Server 2003

Database Server: Microsoft SQL Server on a different machine

Scenario #2 – Single AgilePoint Server Concurrent User Scalability

A large organization was looking to deploy AgilePoint on a server with up to 60,000 end users.

Result at Peak: Simulated 10K user base with 1,152 concurrent users, achieved 10,012 concurrent processes without reaching a capacity peak.

Hardware configuration: Pentium machine with 3.0 GHz, 4GB RAM, on 2 Processors

Operating System: Windows Server 2003

Database Server: Loaded on the same machine as the AgilePoint BPM suite

Benchmarking

The biggest volume of process instances currently is at a company called China Trust Bank. Their profile is as follows:

- 8000-11000 process instances/day, 3.6 million process instances/year
- 300-400 concurrent users
- Approximately 80 process templates, average 40-70 steps, most activities are manual tasks and database access AgileParts

Vandemoortele, a Belgian food group with branches in more than 12 European countries, averages 200 users on their systems.

BDO The Netherlands consists of about 90,000 users in the total domain, with an average of 100-200 users connecting at the same time. They are moving towards 400 Envision users.

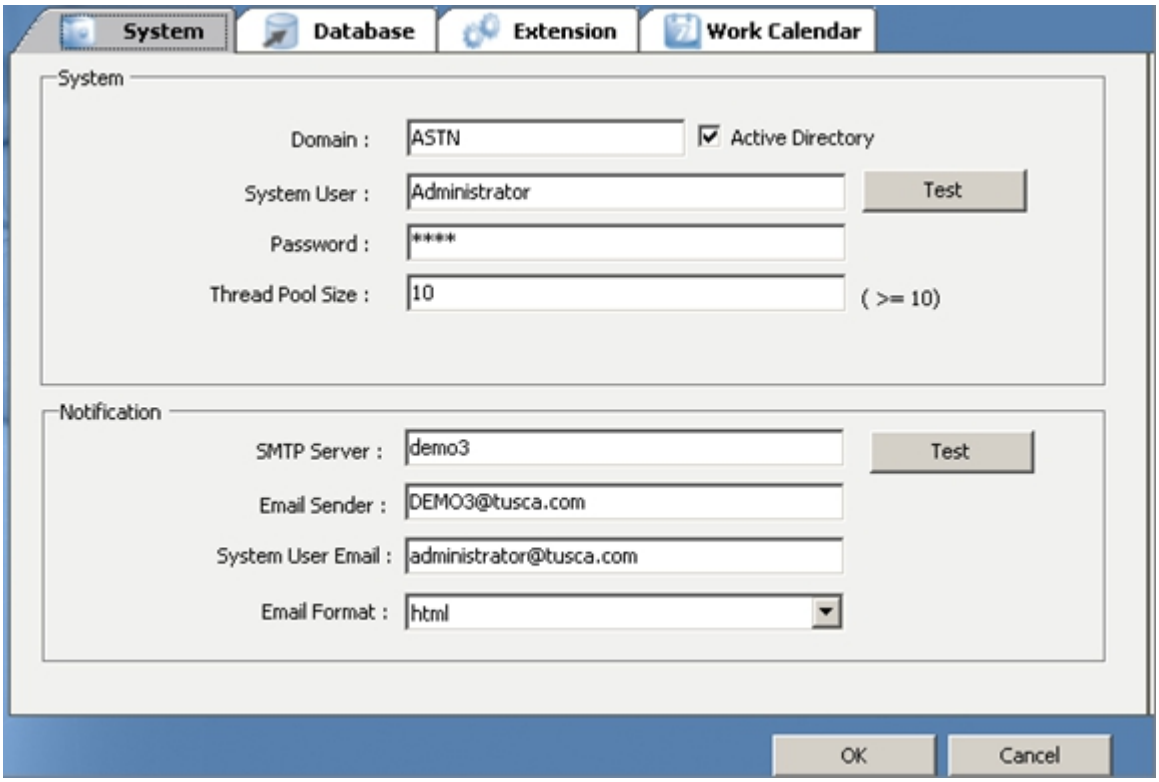
AgilePoint Server Configuration Settings

The factors discussed in this section are important for configuring AgilePoint Server for performance.

Configuring the AgilePoint Server Thread Pool Size

The **System** tab of the AgilePoint Server Configuration utility provides the user interface to configure the maximum number of threads that AgilePoint will be able to use at runtime while handling workflow requests. When the thread pool is configured via this utility it sets both the Event Capacity (for processing AgileWorks) as well as the Working Capacity (for processing AgileParts). In most cases, the same thread pool size as configured via the utility is sufficient for both the Working Capacity as well as the Event Capacity.

For thread pool sizing, anything between 10-100 should be fine. In most cases, a thread pool size of 10 provides a good balance between concurrent processing and server resource optimization.



In some cases, additional tweaking of thread pool sizing may improve performance when heavy CPU usage is apparent for processing AgileParts versus AgileWorks. This finite control over thread pool sizing can be accomplished via the AgilePoint Server Configuration file (i.e., netflow.cfg).

As stated above, when using the AgilePoint Server Configuration utility to set the Thread Pool Size, the value entered, sets both the Event Capacity (for processing AgileWorks) as well as the Working Capacity (for processing AgileParts). However, manual customization can be done via the netflow.cfg to adjust the Working Capacity for a thread pool without changing the Event Capacity or vice versa. For example, if your CPU is processing a large number of AgileParts rather than AgileWorks, you may want to increase the Working

Capacity thread pool size, and leave the Event Capacity as is. Improving AgilePart performance is the key to improving capacity because AgilePart processing can cause a bottleneck.

To modify the netflow.cfg file for thread pool optimization, follow these steps.

1. Open the **netflow.cfg** file and locate the server **eventCap** and **workingCap** attributes.
2. Increase the **eventCap** number to increase the available threads for processing AgileWorks or increase the **workingCap** to increase the number of threads for processing AgileParts. The results are viewed via Enterprise Manager on the Performance page.

Thread Pool Sizing Guidelines

When configuring thread pool size, bigger is not always better! More concurrently running threads requires more server resources (e.g. RAM and CPU), so increasing the thread pool size past a certain point could start to decrease performance rather than improve it. In most cases, a thread pool size of 10 provides a good balance between concurrent processing and server resource optimization. Anything between 10-100 should typically be fine for the Thread Pool. Also, it is often just as (if not more) important to configure the IIS App Pool Queue Length Limit appropriately as to fine tune the Thread Pool size.

If all of your process' activities are "short-duration" ones, then you may not need many threads, because the only way that "short-duration" activities alone would typically cause AgilePoint to (simultaneously) utilize the maximum number of threads in the Thread Pool is if a large number of such activities were entered/activated nearly simultaneously. However, if you have a mixture of "long-running" activities and "short-duration" activities, then increasing thread pool sizes may help considerably because "short" activities that start in the "middle" of a long running activity will not have to wait for it to end before being processed. In general, if you analyze a given time period, then increasing either the quantity or the average duration of activities that are active within that time period will increase the likelihood that some of those activities will require concurrent processing (and therefore require a larger Thread Pool size). However, in certain environments, usage patterns may also have a significant affect, and may render some of the above guidelines less (or more) accurate.

In the above context, a "long-running" activity is defined as one that will require server resources for a length of time long enough that it may overlap with other/subsequent activities or processing. Examples of long running activities could include: synchronous AgileParts that require several minutes (or even hours) of processing to complete; Web Service activities with a large timeout value; AgileShapes that interact with (and may have to wait for data from) other servers or systems; etc.

Conversely, a "short-duration" activity is one that will only consume server resources for a brief period. Examples of such activities include: AgileParts that perform minimal processing; Manual activities (or AgileWorks that perform minimal server-side processing); Single- or Multiple-Condition AgileShapes; etc. Note that Manual activities are normally considered to be "short-duration" activities regardless of the activities' time span or due date, because the AgilePoint server's resources are not being used during most of the activities' duration.

Another way to consider the issue is to determine how frequently new processes will be initiated/launched, how long each process takes to complete, and how much of that time is spent on "automatic" activities that are performed by the server. If you have many long-running processes running simultaneously, and each one contains automatic activities that require minutes (rather than seconds or microseconds) to complete, then you are much more likely to have simultaneous workflow processing occurring, which would require more threads for optimal performance.

See below for a list of guidelines that should be helpful in determining a good Thread Pool size for most environments.

Increasing Thread Pool Size

It may help to increase the Thread Pool size if:

- The average server-side processing duration for activities increases.
- The number of activities entered/activated per day increases.
- The number of new process instances per day increases (because this will typically increase the number of activities entered/activated per day).
- The average number of activities per running process instance increases (i.e. if you start deploying much more complicated process models than you were, it increases the likelihood of concurrent processing).
- The average duration (from Start to Complete) of your process instances increases (because this may increase the number of activities entered/activated per day). This is especially true for processes whose duration is determined by the number of times the process instance "loops" back and re-enters an activity earlier in the process model. It may not be relevant for processes whose duration is primarily determined by the durations of the Manual activities in the process.
- Any of your commonly-used processes contain activities that require minutes (or longer) of server-side processing (as described above), as this significantly increases the likelihood of concurrent processing.
- Your environment is prone to intermittent periods of much-higher-than-average load during certain times of day (or week, month, etc.). For example, if your process instances are often started nearly simultaneously (e.g. at the start of the work day or shift, or because of certain events like the upload of a large batch of files), or you have many activities that "time out" simultaneously (e.g. at the end of the month), then it may make sense to increase the Thread Pool size to take the higher-load periods into account.
- Your server, network, or database is prone to intermittent "slowdowns" (i.e. periods of slower-than-normal processing). E.g. This could be due to scheduled services running in the background at specified times of the day, or could be caused by periods of high server load caused by non-AgilePoint sites or applications (on a non-dedicated server).
- Your server has plenty of memory (i.e. RAM), but not a lot of processing speed (i.e. CPU speed). In other words, activities will take longer to process on a slow machine.

Decreasing Thread Pool Size

It may help to decrease the Thread Pool size if:

- Your server has limited memory (i.e. RAM). Regardless of CPU speed, a server with limited RAM should limit the amount of concurrent processing performed (i.e. threads used) in order to limit memory usage.
- All (or almost all) of your processes use only short-duration activities. In such a scenario, it may be faster for the server to process all activities sequentially (more or less) using a small number of threads, than to concurrently process the same activities. In such a scenario, the server resources required in order to split the server's CPU cycles between multiple threads may actually be (slightly) slower than simply waiting for any activities currently being processed to be completed.
- The opposite of any of the items in the previous list is true. However, note that decreasing the Thread Pool size will rarely result in dramatic performance improvements, so you typically don't need to fine-tune the Thread Pool size below 50.

Configuring the Swap Out Time

The Swap Out Time configuration option refers to the maximum amount of time a process instance remains in idle before it is swapped out of memory. The maximum swap out time is set to 5 minutes (300 seconds) by default. Increasing the swap out time lets the process stay in memory longer, avoiding the CPU intensive

swapping, but keep in mind this may eventually affect capacity. Because swapping consumes a lot of CPU time, it may be beneficial to reduce this number if the CPU is low on memory to improve performance.

To change this setting, open the file **[AgilePoint Server installation folder]\netflow.cfg** in Notepad, change the value for swapoutTime to a higher/lower value depending on the needs, then save the file. (Any changes to netflow.cfg require an IISReset in order to take effect.)

Adjusting the swap out time should only be a temporary workaround solution. If performance is an issue, investing in another CPU, more memory, or a faster hard drive, is a better solution.

Database Sizing and Configuration

This section provides guidelines for optimizing your database sizing and configuration.

Database Sizing

AgilePoint supports SQL and Oracle, has data stores, and we typically suggest you to employ your own corporate application database standards regarding initial size and security related to data store.

All information related to the audit trail of a process and all activities/tasks within are stored in the workflow database. The information related to the front-end side is NOT stored in the AgilePoint database. AgilePoint is able to save all information (audit trail) of completed processes into an archive database.

The recommended best practice for data storage is to develop your application's data model as you would normally (with full indexing, relationships, constraints, etc.) as a separate database from the workflow database. You would then store only the "workflow-related" portions of the data in the workflow database (as custom process attributes), so that the data can be used for such purposes as controlling the process flow, building email notifications, etc.

In general you can determine your data store capacity based on the following parameters:

- Number of unique process templates
- Average number of activity steps per process definition
- Size of attachments
- Average size of custom attributes per process (application data carried along the process)

The following database sizing formula can be used to estimate the space required for the AgilePoint Workflow Database.

Assumptions:

- Assume a fixed-size process
- Assume 50 steps are in the process

Database estimates:

Process Instance Table = 5016 bytes

Sum of above = 145 KB (As a general rule the size is generally < 60 KB)

Database size formula in KB: (145 + [workflow relevant data])*(number of processes)

Assumptions:

- Assume average workflow relevant data = 1024 KB
- Assume process instances = 100/day or 36500/year

Total database sizing estimate for a year: 1169*100*365 = 42668500 KB or 42 GB

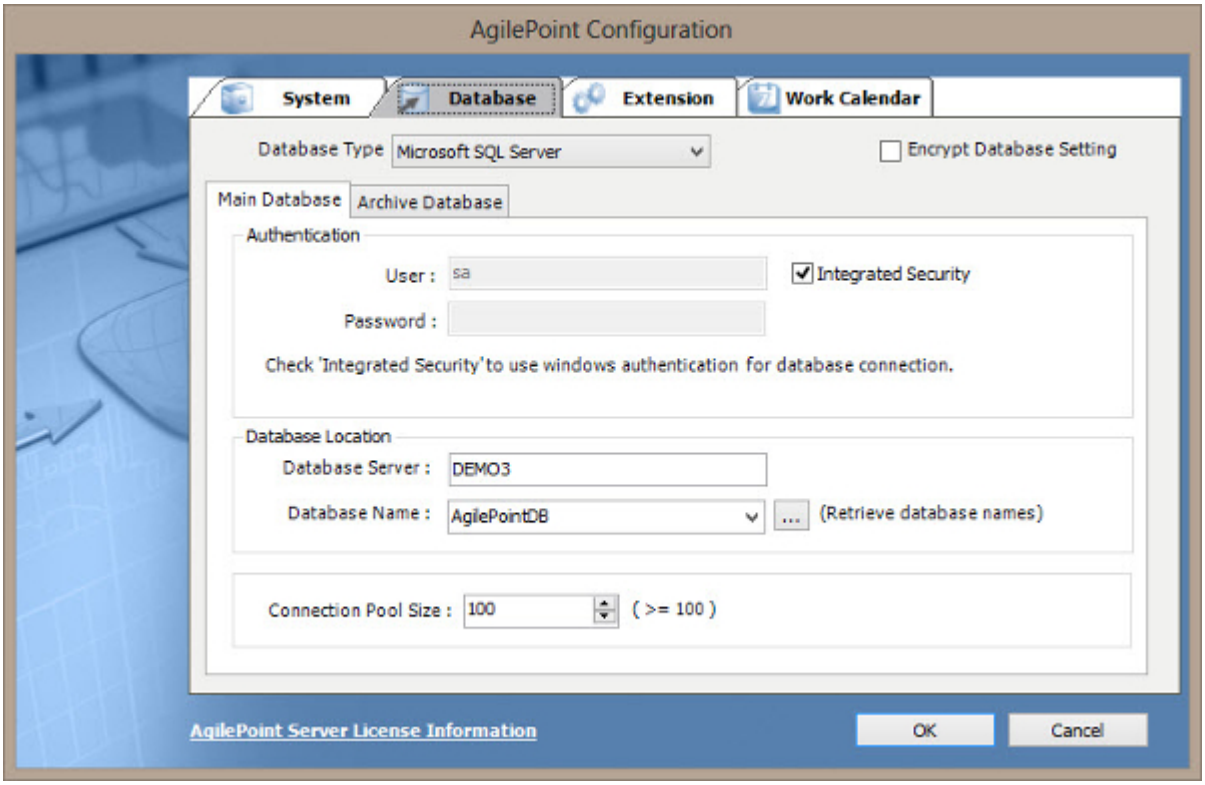
The workflow relevant data is a big percentage of the estimate and varies depending on the application.

Workflow Relevant Data: [WF_CUSTOM_ATTRS; WF_LARGE_BINARIES; WF_LARGE_TEXT]

Typically you should monitor the above parameters in your development and test environments and be able to use the above data for calculating database sizing requirements for your production environment.

Database Archiving

Although AgilePoint is designed to handle large quantities of workflow data efficiently, the amount of data stored in the workflow database may affect the runtime performance of certain operations. As the AgilePoint workflow database grows in size over time, it may improve the system's performance to "archive" some of the older data by moving it into a separate database. This can be done automatically by AgilePoint simply by configuring the Archive Database settings in the AgilePoint Server Configuration utility.



It is good practice to consider archiving completed process data for better performance of database indexing and interactivity. It is recommended to have archiving rules across all process definitions or per process definition. In general, archive process data based on the following parameters:

- Based on corporate reporting requirements for a particular process
- Based on your database hardware
- Based on average transaction response time

You can obtain the archived data from the AgilePoint Archive database by using SQL. Contact AgilePoint Support if you would like to obtain a copy of the AgilePoint Database Schema.

AgilePoint provides the functionality to turn on/off process archiving to the archive database via the AgilePoint Server Configuration Utility. To avoid having your process data archived to the archive database, simply turn off the archiving. To turn off the archive database process, set the "Archive completed process in" field = 0. Risks of not archiving include pile up of records leading to decreased performance of the database indexing and interactivity.

APADM Command Line Utility

The APADM application is a command line utility that can be used to perform several different administrative tasks from the command line.

The APADM command line utility is located in your AgilePoint Server folder at:

```
…\AgilePointServer\Tools\apadm.exe
```

APADM Archive Database Command

First, the utility can be used to manually initiate archiving of AgilePoint Server data to the archive database. Archiving the workflow data will move the data for "old" processes from the main workflow database to an "archive" database. "Old" processes are those that have been Canceled or Completed for at least a specified minimum number of days.

In order to use the utility to archive workflow data, you must have already configured a connection to an archive database using the AgilePoint Server Configuration utility.

In order to use the utility to archive workflow data, you must be logged in to Windows using a user account that has the **Archive Process Instance** AgilePoint access rights.

Below is how to run the command line utility:

```
apadm -archive [days] [AgilePointServer URL]
```

Below is an example of using the utility to archive all processes that were completed or canceled at least 170 days ago:

```
D:\>apadm -archive 170 http://[hostname]/AgilePointServer
Connecting AgilePoint Server...You currently logon as 'MYDOMAIN\admin'
Query AgilePoint database...12 process(es) can be archived.
Please enter <Y> to continue, otherwise stop...y
Archiving process '/sites/AgilePointFinancialSite/ExpenseReport_InfoPath/
ER-2005-0032.xml#1'…Done ( 0.334(s))
```

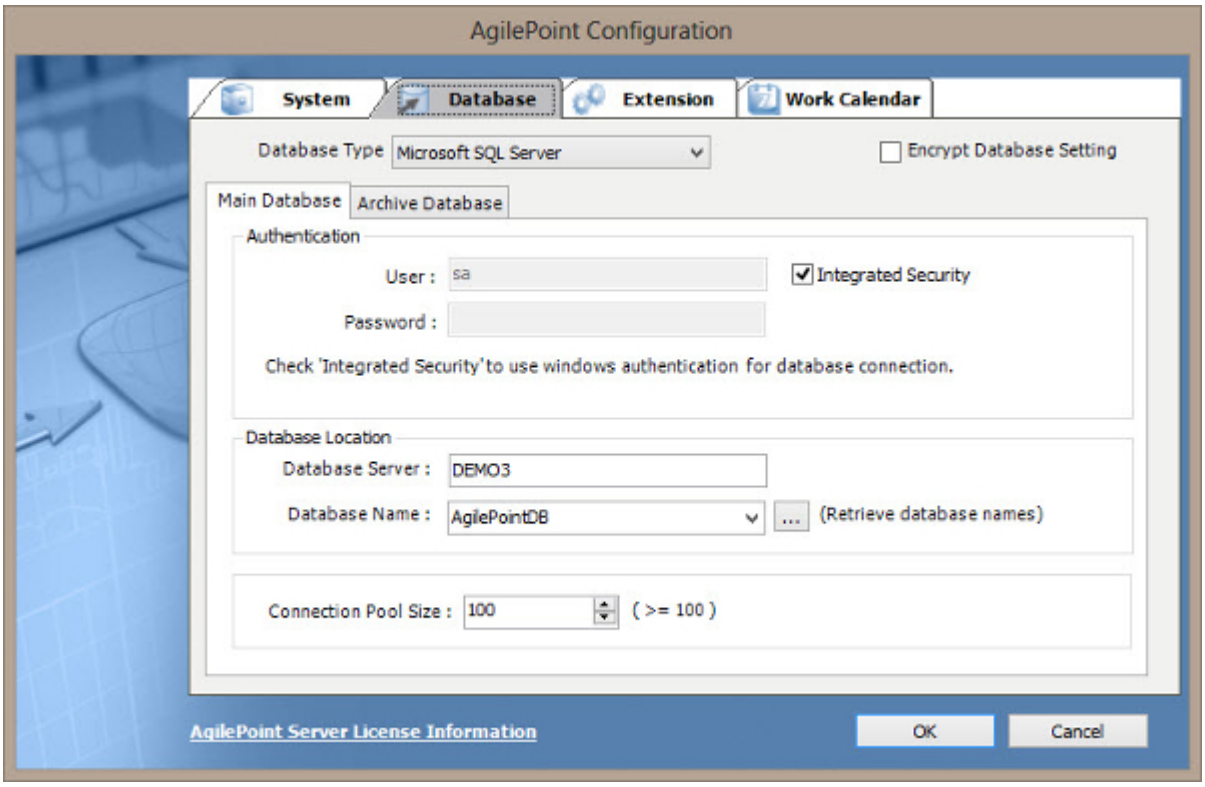
Also you can create a Windows scheduled task to automate the process with your own schedule using this command.

Database Connection Pool Size

AgilePoint Server allows you to configure the maximum number of "database connections" that it will use to connect to the workflow database at runtime.

The optimal value for this setting should usually be based on the value of the Max Thread Pool Size setting. In almost all cases, the database connection pool size should be at least (2 * the thread pool size). On average, 20 database connections can support 10000 processes on a single physical machine.

The Database area of the AgilePoint Server Configuration utility provides the user interface to configure the database connection pool size.


AgilePoint Configuration
System
Database
Extension
Work Calendar
Database Type
Microsoft SQL Server
Encrypt Database Setting
Main Database
Archive Database
Authentication
User :
sa
Integrated Security
Password :
Check 'Integrated Security' to use windows authentication for database connection.
Database Location
Database Server :
DEMO3
Database Name :
AgilePointDB
...
(Retrieve database names)
Connection Pool Size :
100
(>= 100)
AgilePoint Server License Information
OK
Cancel

Client Applications

Client applications can communicate with AgilePoint BPMS Engine in different ways via HTTP/WebDav (Web Services), Remoting, WCF, Messaging (JMS/MSMQ) which provides flexibility in fine tuning performance needs. AgilePoint also supports client applications to run in their native environments and communicate with the BPMS Engine. This offers users great flexibility in scenarios such as front end Java Web Applications running on their enterprise hardware and software finely tuned to handle heavy volume retaining their efficiency while interacting with the BPMS Engine.

We recommend the following guidelines for client applications interacting with AgilePoint Server.

- **Web Service** - We recommend this as the default option for client applications communicating with AgilePoint as it is the de facto standard of the SOA paradigm.
- **MSMQ/JMS** - We recommend this approach in scenarios where client applications want to interact in loosely coupled mode via messages. Also they want messages to be delivered in persistence mode.
- **.NET Remoting** - We recommend this approach in scenarios where integrated applications want to have high performance data transferring between AgilePoint Server and Client Applications.

Tips for Improving Performance and Troubleshooting

In addition to the topics already covered in this document, this section describes some additional specific usage patterns and issues that may affect runtime performance and server resources.

Out of Memory Errors

Generally the most common performance problems are related to Out Of Memory errors. Out Of Memory errors can occur at any location in the code, therefore, you need to eliminate those in order to determine if there are any separate issues.

To begin troubleshooting for Out of Memory errors, do the following:

1. Open the AgilePoint Server Configuration and reduce the Thread Pool Size to (e.g. 50) and the Database Connection Pool Size to (e.g. 100). If that doesn't eliminate the Out Of Memory errors, try reducing the sizes even further. Remember that increasing pool and queue sizes to increase multithreading only improves performance up to the point that the server's hardware can support the concurrent processing. Beyond that, it becomes detrimental to both performance (and potentially to server stability).
2. Examine what other applications are running on the server and try to free up as much of the server's memory as possible for AgilePoint's use.
3. Ensure there are no other applications using the Application Pool for AgilePoint Server.
4. Out Of Memory errors could be due to an overloaded Database server, poor network performance, or other reasons. Check the Database server's stability and logs to see if anything is wrong with it. If the Database server is a performance bottleneck, it could potentially cause in-process work to build up on the AgilePoint server. To resolve this issue, try moving from a 32-bit to 64-bit server, this will also enable the advantages of table partition.
5. If the above steps don't resolve the Out Of Memory errors, try increasing the amount of physical memory, move from 32-bit to 64-bit server, or change other server configuration settings to prevent the server from becoming overloaded.

An expired trial license could also cause problems, because it may have caused a large amount of work to build up waiting for server processing. Once the key was replaced, if the server tries to concurrently process too much work (because of the configured settings), it could potentially cause Out Of Memory errors.

During very high load, server's may not recycle "unused" memory as quickly as at other times, so servers that have periods of very high load and high-memory allocation may run out of memory simply because they use up (i.e. allocate) memory faster than the server can recycle (i.e. de-allocate) the memory and make it available for use again.

Note: The amount of memory displayed by a Windows process in the Windows Task Manager does not necessarily reflect the amount of memory currently needed (or even being used) by that Windows process. If you are trying to measure how much memory a .NET-based application is actually using (i.e. the application's memory footprint) or allocating, you should see one of the many tutorials available online about how to do so accurately. There are also many products available (both commercial and free) specifically designed to assist in memory analysis.

Virtual Memory

Beyond the limit of the server's physical memory, the system will try to use virtual memory (i.e. a portion of the server's hard disk will be used as additional RAM). Since hard disks are drastically slower than RAM, this should be avoided as much as possible. Ways to prevent unnecessary virtual memory usage include:

- Avoid running unnecessary applications on the server (e.g. fancy screen savers)
- Try to run "scheduled" services (e.g. virus scans, backups, or batch uploads) during times when the server will be under minimal load (this will be dependent on your system's actual usage patterns)
- For non-dedicated servers, consider whether moving some of the server's functionality to a dedicated server would improve performance

Affect of Customizations on Server Resources and Performance

While custom extensions of AgilePoint are technically covered by the list above, it should be noted that certain AgilePoint customizations or extensions may affect server resource usage and performance more than others. For example, some custom AgileShapes, custom report queries, or Workflow API calls may require very little server resources to complete, while other implementations of the same or similar functionality might require significantly more time and resources to process. This is no different from any other application integration scenario, where the exact implementation of a particular requirement or feature will affect the actual performance of the system. However, it is specially noted here to draw attention to the fact that the performance affects of AgilePoint customizations cannot be entirely predicted or controlled by AgilePoint itself. Therefore, developers who are implementing such customizations should try to take into consideration any potential affects that their customizations may have on the overall performance of the AgilePoint system.

Hard Disk Fragmentation

As with any computer system, fragmentation of the files and/or white space on the server's hard disk(s) can potentially cause very severe performance issues. Therefore, it is a good idea to periodically defragment your servers' hard disks to avoid any such potential issues.

Scheduled Tasks, Screen Savers, etc.

As previously mentioned in the hardware section (above), it is a good idea to avoid running unnecessary applications, services, etc. on your servers. Also, when scheduling necessary system tasks and operations (e.g. backups, defragmentation, virus scanning, etc.) it is best to attempt to schedule them for times and days when the server is expected to be under minimal load by users, so as to avoid affecting the users' perceptions of the system's performance while the scheduled operations are running.

File Locking, Indexing, and Scans

In addition to the issues previously mentioned regarding scheduled tasks, there is an additional issue to consider regarding operations that may lock (or otherwise prevent access to) files on the file system. Common examples of such operations include:

- Microsoft Indexing Service
- Microsoft File Replication Service (FRS)
- Anti-Virus software

Such applications and services may "lock" files that they are scanning while they are scanning them. (Some have even been known to fail to release the file locks afterwards.) This can potentially prevent AgilePoint or other applications from accessing and/or updating critical files if such files are locked when AgilePoint attempts to access them.

Therefore, the server's administrator should carefully consider whether to enable such applications and services, the times that scans should be scheduled to run, and whether any exclusions or other configuration of the scanning software are necessary.

Anti-Virus Software, Firewalls, Pop-up Blockers, Proxies, etc.

In addition to the issues previously mentioned regarding scheduled tasks and scanning software, there are some additional issues to consider regarding anti-Virus software, firewalls, pop-up blockers, etc. First is that viruses, spy ware, etc. can drastically affect the performance and even stability of your systems and data, and therefore it is very important that you protect your servers with virus protection software, firewalls, etc. The considerations listed previously should in no way be considered as a recommendation not to use such security measures.

However, there are many different varieties of such security software, and some of them may interfere with legitimate network traffic or server functionality. Usually, such interference results in outright runtime errors rather than causing poor performance. However, depending on the implementation and configuration of the specific software or security system used, either performance issues and/or failure of important runtime functionality could potentially occur.

AgilePoint Development Considerations

Separating your ASP.NET application and AgilePoint Server will definitely help by improving the performance of your ASP.NET pages. In addition, improving AgilePart performance is the key to improving capacity because AgilePart processing can cause a bottleneck. Manual activities generally have to wait for human action, which consumes CPU resources so processes should be analyzed for manual intervention that can be automated.

If this has not been done already, it is recommended to upgrade ASP.NET 1.1 to ASP.NET 2.0 for both AgilePoint Server and front end ASP.NET applications.

Windows and Pages that Refresh Automatically

It should be noted that certain windows and pages displayed by AgilePoint automatically refresh the window's contents at regular intervals. Examples of this include:

- The Process Monitor dialog/page displayed by the AgilePoint SharePoint Integration component's Task List Web part
- The Process Monitor dialog/page included as part of ASP.NET applications built using the AgilePoint ASP.NET Web Application project template (included with the AgilePoint Developer component) and referenced by the AgilePoint Task List web control (also included with AgilePoint Developer)
- The Process Monitor included as part of the AgilePoint Enterprise Manager component

This automatic refresh behavior is a usability feature that prevents users from having to manually refresh the page to see if changes have occurred. However, since such pages interact with the server both when first displayed, and also at recurring intervals (if the window remains open), those pages do represent a potential source of unnecessary load for the server if the pages are left open by users unnecessarily. When they are left open unnecessarily, the windows may still consume server resources in much the same way that an active user interacting with the system would do so. If such server load starts affecting overall server performance, then it might be necessary to alter the system's configuration to decrease the default refresh frequency (or even prevent it altogether) in order to ensure optimal system performance.

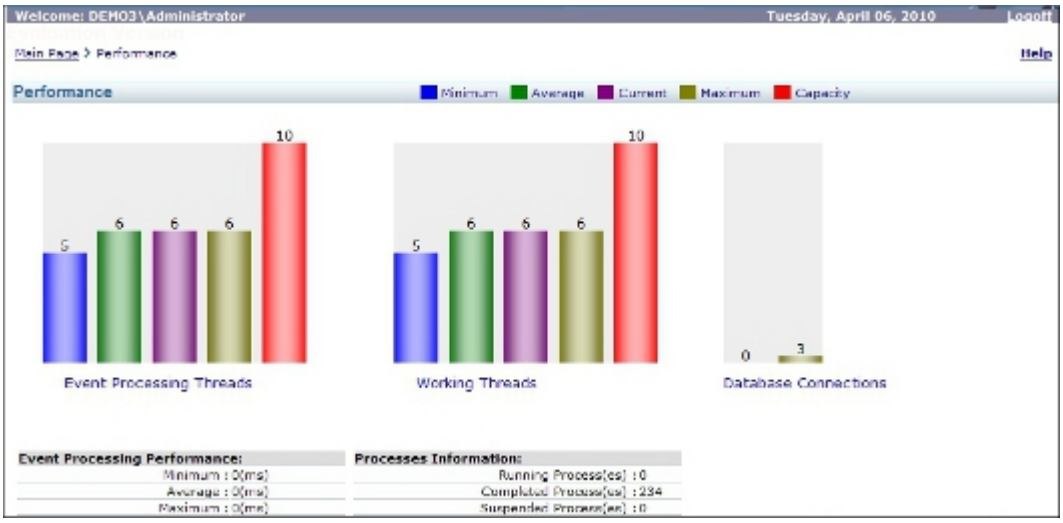
Task List Control Placement

Certain AgilePoint components include Web parts and/or ASP.NET controls that can be placed in various and/or multiple locations by a system administrator and/or developer. Some of these, like the AgilePoint Task List controls interact with one or more servers (e.g. AgilePoint, SharePoint, and/or database servers) whenever they are viewed or refreshed. This interaction is necessary and unavoidable in most cases, and is part of the AgilePoint system's expected usage patterns.

However, in some environments, the location in which such a control is placed could have unintended affects on the system's performance. For example, in an environment where there are many thousands of SharePoint users, but only a fraction of those users interact with AgilePoint, then placing an AgilePoint Task List control on an entry or portal page that is frequently accessed by the entire group of SharePoint users may place additional unnecessary load on the AgilePoint system when the page is accessed by users that don't need access to the control. In such a scenario, it may be better to place the control in an alternate location that is only accessed by a more appropriate group of users and/or that is only accessed when necessary.

Database Connection Sizes

The AgilePoint Enterprise Manager Performance page as shown below allows the administrator to monitor the database connections to the workflow database at runtime. If the maximum number of "database connections" has reached or is close to reaching the maximum capacity, this means that either AgilePoint Server or an AgilePart holds the database connection too long.


Welcome: DEMO3\Administrator
Tuesday, April 06, 2010
Logoff
Main Page > Performance
Help
Performance
Minimum
Average
Current
Maximum
Capacity
10
5
6
6
6
Event Processing Threads
10
5
6
6
6
Working Threads
0
3
Database Connections
Event Processing Performance:
Minimum : 0(ms)
Average : 0(ms)
Maximum : 0(ms)
Processes Information:
Running Process(es) : 0
Completed Process(es) : 234
Suspended Process(es) : 0