

Word Automation Services 2010 Capacity Planning Guidance

This document is provided “as-is”. Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

Some examples depicted herein are provided for illustration only and are fictitious.  No real association or connection is intended or should be inferred.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

© 2010 Microsoft Corporation. All rights reserved.

Word Automation Services 2010 Capacity Planning Guidance

Sean Azlin, Chris Vincent

Microsoft Corporation

March 2010

**Applies to: Word Automation Services in SharePoint Server 2010**

Summary: This article contains capacity planning guidance for Word Automation Services 2010. Use this article to help estimate hardware and Microsoft® SharePoint® Server 2010 farm requirements when Word Automation Services usage is desired.

Table of Contents

[Contents 2](#_Toc259010972)

[Test farm characteristic 2](#_Toc259010973)

[Workload 2](#_Toc259010974)

[Hardware settings and topology 3](#_Toc259010975)

[Dataset 5](#_Toc259010976)

[Test results 6](#_Toc259010977)

[Recommendations 11](#_Toc259010978)

[Single server farm 11](#_Toc259010979)

[Basic Word Automation Services farm 11](#_Toc259010980)

[Advanced topologies 12](#_Toc259010981)

[Estimating throughput targets 20](#_Toc259010982)

[Troubleshooting performance and scalability 21](#_Toc259010983)

## Test farm characteristic

### Workload

Testing for Word Automation Services was designed to help develop estimates for how different farm configurations respond to changes in the following variables:

* Number of Word Automation Services-enabled application servers in the farm
* Number of active conversion processes per Word Automation Services-enabled application server
* Number of items in the Word Automation Services database

It is important to note that the specific capacity and performance figures presented in this article will be different from the figures in real-world environments. The figures presented are intended to provide a starting point for the design of an appropriately scaled environment. After you have completed your initial system design, test the configuration to determine whether your system will support the factors in your environment.

#### Test definitions

This section defines the test scenarios for this article and provides an overview of the test process that was used for each scenario. Detailed information such as test results and specific parameters are given in the Test Results sections later in this article.

|  |  |
| --- | --- |
| Test name | Test description |
| Throughput Scale | 1. Create a SharePoint library and populate it with some number of valid OpenXML files (.docx) 2. Create and start a conversion job using the library from step 1 as an input library. 3. When the conversion job is complete (that is, all conversion items have succeeded or failed), use the results in the Word Automation Services database to determine the overall throughput of the service when conducting the conversions. |
| SQL Server Database File Size | 1. Create a SharePoint library and populate it with some number of valid OpenXML files. 2. Start and cancel conversion jobs to populate the database. Allowing the conversion jobs to complete is not necessary. 3. Record the size of the database LDF and MDF files. |

Table 1 – Test definitions for this article

### Hardware settings and topology

#### Lab hardware

To provide a high level of test-result detail, several farm configurations were used for testing. Farm configurations ranged from one to seven application servers and a single database server that is running Microsoft SQL Server® 2008 database software. All servers were 64-bit.

The following table lists the specific hardware that was used for testing.

|  |  |  |  |
| --- | --- | --- | --- |
| Computer name | Front-end Web server/application server 1 | Application server 2 - 7 | SPSQL |
| Role | Front-end Web server + application server (shared) | Application server (dedicated) | SQL Server cluster (one computer) |
| Processor(s) | 2px4c@2.33 GHz | 2px4c@2.33 GHz | 4px4c@3.2 GHz |
| RAM | 8 GB | 8 GB | 16 GB |
| Operating System | Windows Server® 2008 SP2 x64 | Windows Server 2008 SP2 x64 | Windows Server 2008 SP2 x64 |
| Storage & its geometry (inc. SQL Server disks configuration) | 6 + 75 + 590 GB | 6 + 75 + 590 GB | 6 + 75 + 460 GB |
| # of NICs | 2 | 2 | 2 |
| NIC speed | 1 gigabit | 1 gigabit | 1 gigabit |
| Authentication | NTLM | NTLM | NTLM |
| Software version | 4762.1000 | 4762.1000 | SQL Server 2008 |
| # of SQL Server Instances | N/A | N/A | 1 |
| Load balancer type | NLB | NLB | N/A |
| ULS Logging level | Medium | Medium | Medium |

Table 2 – Lab hardware details for the Word Automation Services test topology

Note that a dedicated front-end Web server was never used for testing. Instead the front-end Web server used to drive testing was also Application Server 1. This is not uncommon for a Word Automation Services-dedicated topology because SharePoint front-end Web servers are not used to process conversions. The only role a front-end Web server would play is to drive the creation of conversion jobs via a custom SharePoint solution (such as a custom Web Part, for example). A front-end Web server might need to remain responsive for a SharePoint solution to work properly.

For the Word Automation Services Test Farm, a simple C# application was used on front-end Web server / application server 1 to occasionally drive the creation of conversion jobs for testing. Maintaining the responsiveness of the front-end Web server was not a concern for this farm, so using the server as an application server was appropriate.

#### Topology



Diagram 1 – Word Automation Services test farm topology

### Dataset

The dataset used for testing comprises 384 unique, OpenXML DOCX files containing the following types of Microsoft Office Word 2007 content:

* Text with Direct Formatting
* Content Controls
* Images
* Tables
* Styles
* Fields
* OLE Objects
* Hyperlinks
* Bookmarks
* Comments
* Citations

These files ranged in size from 20 KB to 8.8 MB, with an average of 225 KB per file. Duplicates of these 384 files were used to create a library of about 20,000 documents. That library was then used as an input library for each test run.

### Test results

The following tables show the test results of Word Automation Services in SharePoint Server 2010. For each group of tests, only certain specific variables are changed to show the progressive impact on farm performance.

#### Throughput scale

##### Effect of Active Conversion Process scale on throughput

###### Overall scale

The two tests in the following table show how the throughput of Word Automation Services increases as the number of active conversion processes are increased gradually on a single application server. Data is shown for two output formats: DOCX and PDF. The DOCX conversions provide a baseline throughput for comparison against other output formats while the PDF conversions provide an example of a more-typical conversion throughput.

|  |  |  |
| --- | --- | --- |
| Active Conversion Processes | DOCX | PDF |
| 1 | 2.72 | 1.13 |
| 2 | 4.65 | 1.78 |
| 3 | 5.92 | 1.99 |
| 4 | 7.02 | 2.00 |
| 6 | 7.73 | 1.87 |
| 8 | 9.45 | 1.64 |
| 16 | 7.91 | 1.41 |
| 24 | 8.06 | 1.37 |
| 32 | 7.71 | 1.37 |

Table 3 – Example throughput of an eight-core application server as active conversion processes are added

Note the decrease in throughput for PDF encountered when using six active conversion processes instead of four. This is due to a per-server limitation in Word Automation Services when converting to PDF (or XPS). In contrast, note that the throughput of DOCX does not have this limitation and continues to increase until eight active conversion processes are used. However, DOCX runs into another, more common limit when the number of active conversion processes exceeds the number of processing cores on the server (which in this case is eight cores).

Also note that the unusually small improvement in throughput for DOCX when using six active conversion processes versus four active conversion processes is a typical variation for Word Automation Services. It’s a good example of how throughput can vary from expectations for a given configuration.

The following is a graph of the above data:

Chart 1 – Example throughput of an eight-core application server as active conversion processes are added

The 16, 24, and 32 active conversion process numbers are shown to drive home the point that having more active conversion processes than there are processing cores is actually detrimental to the throughput of an application server. Conversion items may also be more likely to fail intermittently when using an unsupported number of **Total Active Conversion Processes** for a given application server.

There are two key takeaways from this data:

1. The best throughput improvements for conversion to PDF occur when scaling the number of active conversion processes from 1 active conversion process to three active conversion processes per server. PDF throughput will actually begin to decrease as more active conversion processes are used somewhere around 4 active conversion processes on any server that has four or more processing cores. This is a limitation of Word Automation Services. The same limitation applies to XPS as well.
2. The throughput improvement for other formats, such as DOCX, can scale very well up to N active conversion processes where N is the number of processing cores on the application server. However, note that the recommended maximum number of **Total Active Conversion Processes** for application servers is N-1 for the same N. This is explained more in the [Recommendations](#_Recommendations) section.

##### Effect of application server scale on throughput

###### Overall scale

The two tests in the following table show how the throughput of Word Automation Services increases as the number of application servers is increased gradually. The number of **Total Active Conversion Processes** was set to ‘8’ for the farm. Data is shown for two output formats: DOCX and PDF. The DOCX conversion throughput is a good representative of most output formats while the PDF conversion throughput is better for representing both PDF and XPS.

|  |  |  |
| --- | --- | --- |
| Topology | DOCX | PDF |
| 1x1 | 9.5 | 1.64 |
| 1x2 | 17.3 | 3.25 |
| 1x3 | 23.1 | 4.81 |
| 1x4 | 32.8 | 6.52 |
| 1x5 | 39.7 | 7.87 |
| 1x6 | 45.9 | 9.50 |
| 1x7 | 52.1 | 11.48 |

Table 4 – Example throughput of farm as the number of application servers is increased

Note that the increase in throughput for both PDF and DOCX remains generally linear for each additional server added, as shown in the following chart:

Chart 2 – Example throughput of farm as the number of application servers is increased

The following tables show how the throughput of Word Automation Services is likely to increase according to the test results shown above:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Topology | DOCX | PDF |  | Topology | DOCX | PDF |
| 1x1 | N/A | N/A |  | **1x1** | N/A | N/A |
| 1x2 | 82.11 | 97.57 |  | **1x2** | 82.11 | 97.57 |
| 1x3 | 61.05 | 95.30 |  | **1x3** | 33.53 | 48.24 |
| 1x4 | 102.11 | 103.66 |  | **1x4** | 41.99 | 35.40 |
| 1x5 | 72.63 | 82.21 |  | **1x5** | 21.04 | 20.73 |
| 1x6 | 65.26 | 99.05 |  | **1x6** | 15.62 | 20.69 |
| 1x7 | 65.26 | 120.54 |  | **1x7** | 13.51 | 20.86 |
| Table 5 – percent throughput increase in terms of single server throughput | | |  | Table 6 - percent throughput increase in terms of Z-1’s throughput where Z is the previous topology’s number of application servers | | |

Note that these numbers are only a sample of how throughput might increase in a given production deployment of Word Automation Services. Some variations in these tables may not be typical for other SharePoint farms.

Also note that because **Total Active Conversion Processes** was set to ‘8’, the PDF results are likely less than what could be expected from these application servers with the proper settings (according to what is observed in table 3). So, the PDF throughput numbers in table 4 could likely be improved significantly by setting **Total Active Conversion Processes** to ‘4’. However, this would undoubtedly decrease the throughput numbers for DOCX, again per the results shown in table 3. The takeaway from these observations is that there is a trade-off to consider when choosing a value for the **Total Active Conversion Processes** setting. The recommended Word Automation Services settings in the [Recommendations](#_Recommendations) section of this article take this trade-off into consideration by providing two separate sets of recommended settings.

Another takeaway from this data is that scaling out is a great way of increasing Word Automation Services throughput for any output format. Note that the linear improvement in throughput that is shown here is not likely to scale infinitely as a topology grows in size. Certain bottlenecks will emerge eventually, such as the SQL Server reaching capacity.

#### SQL Server database file size

##### Database size

The Word Automation Services database requires between 1.58 – 0.15 KB of disk space per conversion item in the database, as the following data shows:

|  |  |  |
| --- | --- | --- |
| Items Added | MDF Size (KB) | KB/Item |
| 2,304 | 3,648 | 1.58 |
| 4,608 | 3,648 | 0.79 |
| 23,040 | 6,720 | 0.29 |
| 46,080 | 10,048 | 0.22 |
| 230,400 | 37,952 | 0.16 |
| 460,800 | 72,000 | 0.16 |
| 1,152,000 | 174,400 | 0.15 |
| 2,304,000 | 345,408 | 0.15 |
| 3,456,000 | 515,392 | 0.15 |
| 4,608,000 | 685,376 | 0.15 |
| 11,520,000 | 1,707,328 | 0.15 |
| 23,040,000 | 3,429,568 | 0.15 |

Table 7 – MDF file size for a varying number of conversion items

The takeaway from this data is that the size of the MDF file increases at an eventual rate of about 0.15 KB for each conversion item that is added to the Word Automation Services database. The first 50,000 conversion items or so are an exception, but the total size of the MDF file is clearly manageable when so few conversion items have been added.

Note that it is generally recommended to not let the Word Automation Services database grow to a size of 2,000,000 conversion items or larger. Otherwise, some Word Automation Services solutions may steadily perform less well as the database grows in size.

##### Deleting items from the Word Automation Services database

Approximately 0.2 – 0.5 KB of disk space is used by Word Automation Services in the SQL Server LDF file for every item deleted from the database. The LDF file is used by SQL Server to maintain recovery data for the Word Automation Services database.

|  |  |  |
| --- | --- | --- |
| Items Deleted | LDF Size (KB) | KB/Item |
| 2,304 | 1,856 | 0.56 |
| 4,608 | 2,624 | 0.44 |
| 11,520 | 2,624 | 0.18 |
| 23,040 | 2,624 | 0.09 |
| 46,080 | 20,416 | 0.43 |
| 69,120 | 20,416 | 0.29 |
| 115,200 | 39,936 | 0.34 |
| 172,800 | 53,248 | 0.30 |
| 207,360 | 53,248 | 0.25 |
| 218,880 | 53,248 | 0.24 |
| 228,096 | 53,248 | 0.23 |
| 230,400 | 53,248 | 0.23 |

Table 8 – LDF file size for a varying number of conversion item-deletions

Note that the size of the LDF file expands at certain intervals due to the autogrow settings of SQL Server. More information on the growth of the LDF can be found [here](http://support.microsoft.com/kb/317375/) if this is an unfamiliar concept.

If left unattended for long, the LDF can grow in size until the SQL Server runs out of disk space. So, decreasing the size of the LDF periodically is something that should be considered for any production farm. Information on how to handle an overly large LDF can also be found [here](http://support.microsoft.com/kb/873235).

## Recommendations

### Single server farm

Word Automation Services can be run on a single server installation of SharePoint Server. This server acts as the front-end Web server, the application server, and the database server for the Word Automation Services database and various SharePoint databases.

However, for production purposes it is highly recommended to not use a single server farm. Word Automation Services, SharePoint, and SQL Server will compete for resources, resulting in inconsistent performance from Word Automation Services.

### Basic Word Automation Services farm

A basic Word Automation Services farm is composed of two servers: a single server to act as both front-end Web server and application server, and a second server to act as an instance of SQL Server for SharePoint and Word Automation Services. Such a configuration should be considered an absolute minimum topology for a production Word Automation Services farm. Expanding beyond this basic topology is explained in increased detail in the next section.



Diagram 2 – Simple Word Automation Services farm topology

### Advanced topologies

To increase the capacity and performance of the basic Word Automation Services farm, you can do one of two things. You can either scale up by increasing the capacity of your existing application servers or scale out by adding additional servers to the topology. This section describes the general performance characteristics and recommended settings of several topologies that combine these two strategies in various ways. Note that not all possible topologies are represented; these are only some select examples.

#### Scaled-out topology 1: more application servers

A scaled out topology increases the capacity of a farm by adding more application servers to the farm. As the test results in table 4 show, this strategy is great for increasing a farm’s capacity for any output format. Scaling out is a great next step when scaling up existing servers will no longer benefit Word Automation Services’ throughput.



Diagram 3 – Scaled-out Word Automation Services farm topology with three application servers

#### Scaled-out topology 2: reducing SQL Server effect

Word Automation Services maintains its own SQL Server database. In a basic Word Automation Services farm, both the Word Automation Services database and the various SharePoint databases exist on the same physical instance of SQL Server. Word Automation Services will impact both SharePoint databases (for example, getting input files from or putting output files to the content database) and the Word Automation Services database (for example, updating the status of a conversion item when a conversion completes successfully).

To prevent a shared database server from becoming a bottleneck for both Word Automation Services and Sharepoint, a separate physical database server can be created to host the Word Automation Services database. This may or may not improve Word Automation Services throughput and reliability depending on if SQL Server is indeed a bottleneck for a given farm.



Diagram 4 - Word Automation Services farm with dedicated SQL Server topology

Note that a single database server is typically not a bottleneck for small farms, especially if Word Automation Services is the only service being used.

#### Scaled-up topology: dedicated Word Automation Services farm

A dedicated Word Automation Services farm is the absolute best topology possible for maximizing Word Automation Services’ throughput. This type of topology involves increasing the capacity of individual servers in the farm by “throttling up” Word Automation Services to fully leverage application server resources. Several key service settings must be properly configured to accomplish this without running into service limits.



Diagram 5 – An example of a dedicated Word Automation Services farm topology

Note that it can be beneficial to run Word Automation Services solutions on a front-end Web server separate from the farm’s dedicated application servers if the solution driving Word Automation Services will be used to create lots of small conversion jobs. In such a case, a dedicated front-end Web server helps ensure that the solution stays responsive even when the application servers are under load. The topology diagram above shows an alternative topology where the front-end Web server is also an application server such that it will still be used for processing conversions. Such a topology could be ideal if the solution driving Word Automation Services will be creating only a few large jobs occasionally.

Dedicated Word Automation Services farms should typically use the following settings:

##### For PDF/XPS output formats

* **Total Active Conversion Processes** is set to N-1 where N is the number of available processing cores in each individual server **OR** ‘4’, whichever value is smaller.
  + **Example:** This setting would be ‘4’ when the farm’s applications servers have two quad-core CPUs, because ‘4’ is smaller than N-1, which is ‘7’.
  + **Note:** As shown in table 3, converting documents to a “fixed” output format such as PDF run into some hard limits of the service that greatly limit the scale up-potential of individual servers for PDF/XPS. ‘4’ cores tends to be the sweet spot for maximizing throughput on an application server when outputting to PDF or XPS. Increasing the value for this setting will actually decrease throughput for PDF/XPS.
* **Frequency with which to start conversions (minutes)** is set to ‘1 minute’.
* **Number of conversions to start (per conversion process)** is set to ‘30’.
  + **Note**: This value enables a maximum RPS of up to 0.5 conversions per second for each active conversion process in the farm. As shown in tables 3 and 4, this is a fairly reasonable RPS to target so that a maximum possible throughput for a farm is achieved for the PDF/XPS output formats.

##### For DOCX, DOC, and other output formats

* **Total Active Conversion Processes** is set to N-1 where N is the number of available processing cores in each individual server
  + **Example:** This setting would be ‘7’ when the farm’s applications servers have two quad-core CPUs.
  + **Note:** It is recommended to never set **Total Active Conversion Processes** toany value greater than N-1. Some reliability and responsiveness problems may begin to occur when a larger value is used.
* **Frequency with which to start conversions (minutes)** is set to ‘1 minute’.
* **Number of conversions to start (per conversion process)** is set to ‘72’.
  + **Note**: This value enables a maximum RPS of up to 1.2 conversions per second for each active conversion process in the farm. As shown in table 4, this is a reasonable RPS to target such that a maximum possible throughput for a farm is always achieved for non-PDF/XPS output formats.

#### Throttled-down topology: production SharePoint farm with shared application servers

Because an active conversion process will use at most one processing core at a time, it is possible to “throttle down” Word Automation Services by setting the **Total Active Conversion Processes** setting to significantly less than the total number of available processing cores for each application server. Throttled-down application servers in this topology will thus always have processing cores free for other tasks or services, which generally helps to prevent application servers from becoming unresponsive when Word Automation Services is under load.



Diagram 6 – An example of a production farm topology with shared application servers running Word Automation Services

By default, Word Automation Services is throttled-down with a **Total Active Conversion Processes** setting of ‘1’. This is expected to be overly conservative for most Word Automation Services deployments, and the following settings should be used on a typical throttled-down topology:

##### For PDF/XPS output formats

* **Total Active Conversion Processes** is set to (N/2)-1 where N is the number of available processing cores in each individual server **OR** ‘4’, whichever value is smaller.
  + **Example:** This setting would be ‘3’ when the farm’s application servers have two quad-core CPUs because (8/2)-1 is 3, and 3 is smaller than 4.
  + **Example:** This setting would be ‘4’ when the farm’s application servers have four quad-core CPUs, because ‘4’ is smaller than (16/2)-1, which is ‘7’.
  + **Note:** By leaving a single processing core unused, the application server is able to remain more predictable even when the Word Automation Services timer job is executing, which can temporarily dominate an additional processing core. This is true for all topologies. So, essentially these settings restrict Word Automation Services to a peak CPU utilization of 50%. To decrease the peak CPU utilization of the service even more, decrease the value of this setting to (N/2)-2, (N/2)-3, etc.
* **Frequency with which to start conversions (minutes)** is set to ‘1 minute’.
* **Number of conversions to start (per conversion process)** is set to ‘30’.
  + **Note**: This value enables a maximum RPS of up to 0.5 conversions per second for each active conversion process in the farm. As shown in table 4, this is a reasonable RPS to target to ensure proper throughput is achieved.

##### For DOCX, DOC, and other output formats

* **Total Active Conversion Processes** is set to (N/2)-1 where N is the number of available processing cores in each individual server
  + **Example:** This setting would be ‘3’ when the farm’s applications servers have two quad-core CPUs.
  + **Note:** By leaving a single processing core unused, the application server is able to remain more predictable even when the Word Automation Services timer job is executing, which can temporarily dominate an additional processing core. This is true for all topologies. So, essentially these settings restrict Word Automation Services to a peak CPU utilization of 50%. To decrease the peak CPU utilization of the service even more, decrease the value of this setting to (N/2)-2, (N/2)-3, etc.
* **Frequency with which to start conversions (minutes)** is set to ‘1 minute’.
* **Number of conversions to start (per conversion process)** is set to ‘60’.
  + **Note**: This value enables a maximum RPS of up to 1.0 conversions per second for each active conversion process in the farm. As shown in table 4, this is a reasonably conservative RPS target for non-PDF/XPS output formats.
  + **Note**: Setting this value to ‘60’ instead of ‘72’ will make it more likely for application servers to have all of their processing cores available for a short period of time (several seconds) per unit of time as set by the **Frequency with which to start conversions (minutes)**, which in this case is ‘1 minute’. This can be advantageous depending on the needs of the farm. Lowering this setting even more will free up all application server processing cores for an even longer period of time, but at an additional cost to throughput.

#### Mixed topology: production SharePoint farm with a mix of Word Automation Services-enabled application servers and other application servers

A mixed topology of Word Automation Services-enabled application servers and non-Word Automation Services-enabled application servers is a great way to get high Word Automation Services throughput without impacting other SharePoint services. Advantages of a mixed farm are:

* Same advantages as a dedicated farm (when using dedicated Word Automation Services servers)
* Other services are minimally impacted by Word Automation Services thanks to some application servers not being used for Word Automation Services at all.

The disadvantage of using a mixed farm:

* More physical servers may be required versus using a shared farm or dedicated farm
* All Word Automation Services application servers will use the same settings

All mixed farms can be roughly configured in one of two ways:

* Non-Word Automation Services application servers are mixed with shared, *throttled-down* Word Automation Services-enabled application servers
* Non-Word Automation Services application servers are mixed with dedicated, *throttled-up* Word Automation Services application servers

A mixed topology might look like this:



Diagram 7 - An example of a production farm topology with application servers that are dedicated to Word Automation Services

The Word Automation Services-enabled servers in a mixed farm can be configured similarly to either the shared application servers in a [Throttled-down Topology](#_Throttled-down_Topology_:) or the dedicated application servers in a [Scaled-up Topology](#_Scaled-up_Topology_:) to enable the same throughput of either of those topologies.

### Estimating throughput targets

To determine the target throughput of a given topology when given specific settings, the following information can be used:

#### Throughput in conversions per minute per application server:

(**Total Active Conversion Processes** \* **Number of conversions to start (per conversion process)**)

**Frequency with which to start conversions (minutes)**

Note that:

* The result of the above equation, if converted to conversions per second per active conversion process, should not exceed 1.2 for non-PDF/XPS output formats, or 0.5 for PDF/XPS output formats. Exceeding these values could lead to decreased throughput and an increased number of conversion failures.
* If the **Frequency with which to start conversions (minutes)** is increased (that is, the Word Automation Services timer job will run less often) and the total throughput of the farm needs to remain the same, then the **Number of conversions to start (per conversion process)** should be increased in direct proportion to **Frequency with which to start conversions (minutes)**.
  + **Example:** Imagine that an admin wants the Word Automation Services timer job to run less frequently, but she also wants the throughput of Word Automation Services to remain unchanged. Here is how the Word Automation Services settings should be changed to accomplish this:

Original Settings:

* + - **Frequency with which to start conversions (minutes)** is ‘1 minute’
    - **Number of conversions to start (per conversion process)** is ‘72’

New Settings:

* + - **Frequency with which to start conversions (minutes)** is ‘10 minutes’
    - **Number of conversions to start (per conversion process)** is ‘720’
  + **Note: Total Active Conversion Processes** should NOT be changed in proportion to **Frequency with which to start conversions (minutes)**

#### Throughput in conversions per minute for an entire farm:

For PDF/XPS output formats:

* Multiply the result of the previous equation by the number of application servers with Word Automation Services enabled. This is roughly in line with the data shown in table 5.

For other output formats:

* Multiply the result of the previous equation by the number of application servers with Word Automation Services enabled. Then, multiply that result by 0.65. The final result will reflect the data shown in table 5.

## Troubleshooting performance and scalability

|  |  |  |
| --- | --- | --- |
| Bottleneck or Issue | Cause | Resolution |
| The throughout when converting to PDF or XPS does not improve with more than 3 or 4 active conversion processes even when more processing cores are available. | Word Automation Services is fundamentally limited in how fast it can convert files to PDF or XPS on a single application server. Specifically, Word Automation Services throughput cannot be made to increase by scaling up beyond three to four active conversion processes per application server. Adding more active conversion processes per application server will actually degrade the performance of the services performance when converting to PDF or XPS format. | If increasing Word Automation Services’ throughput is necessary for conversions to PDF and XPS even when the number of active conversion processes per application server is set at three to four, then adding additional application servers will yield a near-100 percent increase in throughput given the same computer specifications.  Also, if the number of active conversion processes per application server is set to a value higher than 4, then changing that value to be 4 will likely increase throughput by a small margin. Note, however, that doing this may greatly decrease throughput for other output formats.  If converting to PDF or XPS is a primary need, then it might be more cost effective to not use dedicated Word Automation Services application servers. Shared servers could be used instead so that spare processing cores are available for other farm services. |
| Conversion items begin to fail more often after changing the settings for Word Automation Services | The settings of Word Automation Services can easily be set to push the service past its practical limits. Doing so will likely result in:   * Less overall throughput * Conversion items will fail more often | Following a few simple rules can help correct or prevent either of the previously listed symptoms:  Never set Total Active Conversion Processes to be more than N-1 where N is the number of processing cores of the application server.  Also, never set **Number of conversions to start (per conversion process)** to be greater than what is suggested in [Scaled-up Topology: A Dedicated Word Automation Services Farm](#_Scaled-up_Topology:_A) for the desired output formats unless **Frequency with which to start conversions (minutes)** is also adjusted (in direct proportion) such that the resulting throughput target remains the same. |
| Other services begin to be less responsive after changing the settings for Word Automation Services | An active conversion process will at times fully leverage a processing core on the application server. An application server configured as recommended in [Scaled-up Topology: A Dedicated Word Automation Services Farm](#_Scaled-up_Topology:_A) may thus use almost the entire CPU at any given time given that the Word Automation Services timer job also runs on its own core periodically.  So, if other services such as Microsoft Office Excel or Microsoft Office Web Apps (for example) require CPU resources from such an application server then there might be an undesirable wait time that results in increased perceived latency for those other services. | A few things can be done to decrease the impact of Word Automation Services on other services:  1) Throttle down the **Total Active Conversion Processes** for Word Automation Services, perhaps to the level recommended in [Throttled-down Topology: A production SharePoint Farm with shared application servers](#_Throttled-down_Topology_:).  2) Add additional application servers that are not used by Word Automation Services but are usable for other services, perhaps adopting a topology similar to that described in [Mixed Topology: A production SharePoint Farm with a mix of dedicated Word Automation Services application servers and other application servers](#_Mixed_Topology:_A). |
| Sometimes conversion items fail with error code 3 when the farm was busy, offline, or undergoing maintenance for part of a day or more. | Word Automation Services generally requires that conversion jobs added to the Word Automation Services queue database be fully processed within 24 hours of being submitted. If conversion items for a conversion job are not finished within 24 hours, then the conversion item may fail with error code 3. The error message would read:  “The file could not be downloaded from the input library because the supplied user permissions expired before the file could be retrieved. This likely indicates that the system is under heavy load. Please try resubmitting the job, and contact your system administrator if the error reoccurs." | If users are consistently seeing this error message and the farm was not offline for a significant period of time, then it’s likely that some conversion jobs are taking longer than 24 hours to complete, probably because the farm is either misconfigured or its usage far exceeds its capacity. This means that it is time to take action to increase the capacity of the farm. This would involve either increasing the **Total Active Conversion Processes (up to N-1 where N is the number of processing cores on each individual application server)** or, when that’s not possible, adding more application servers to the farm that Word Automation Services can use. The latter might involve simply enabling Word Automation Services on some application servers that otherwise are not being used much, or it may involve adding more physical servers to the farm.  Whatever is done, be sure that the settings for Word Automation Services are configured correctly per either [Scaled-up Topology: A Dedicated Word Automation Services Farm](#_Scaled-up_Topology:_A) or [Throttled-down Topology: A production SharePoint Farm with shared application servers](#_Throttled-down_Topology_:). |
| The execution time of a Word Automation Services solution takes progressively longer to run as the service has been running. | The execution time of the following Word Automation Services Object Model methods scales with the number of items in the Word Automation Services database:   * ConversionJob.GetAllActiveJobs * ConversionJob.GetAllJobs | It is recommended to not allow the Word Automation Services database reach a size of over 2,000,000 conversion items. Delete some conversions items from the database to correct this problem. |
| Scaling out the number of application servers has stopped improving Word Automation Services’ throughput. What could cause this? | If adding more application servers to the farm is no longer improving throughput, then this could be a sign that the instance of SQL Server that the Word Automation Services database lives on is at capacity.  Some data around Word Automation Services SQL Server impact for each WAS Action is as follows:   |  |  |  |  | | --- | --- | --- | --- | | WAS Action | RT Per call | Additional RT Per Item | Notes | | ConversionJob.AddFile | 11 | 0 | Low SQL Server IOps | | ConversionJob.AddFolder | 9 | 2 | Despite the increase in SQL Server RT versus AddLibrary, this OM call is generally faster to execute than AddLibrary  Low SQL Server IOps | | ConversionJob.AddLibrary | 4 | 2 | Low SQL Server IOps | | ConversionJob.Start | 3 | 0 | Low SQL Server IOps | | ConversionJob.Refresh | 1 | 0 | Low SQL Server IOps | | ConversionJob.CancelJob | 1 | 0 | Low SQL Server IOps | | ConversionJob.GetAllActiveJobs | 1 | 0 | SQL Server IOps scale with the number of jobs in the Word Automation Services DB SQL Server IOps scale with number of jobs in DB | | ConversionJob.GetAllJobs | 1 | 0 | SQL Server IOps scale with the number of jobs in the Word Automation Services DB | | ConversionJob.GetItems | 2 | 0 | Low SQL Server IOps | | Each Timerjob Run | 2 | 1 | Low SQL Server IOps |   Note that some calls have a constant overhead in terms of SQL Server round trips per action and additional SQL Server round trips per action depending on the number of conversion items involved. Also note that most Word Automation Services actions  This information can be helpful for both developers who are planning to create custom solutions for Word Automation Services deployments and also farm admins who must plan for the SQL Server impact of using Word Automation Services | If the Word Automation Services SQL Server database lives on the same server as other active databases, then giving the Word Automation Services SQL Server database its own physical server should remove SQL Server as a bottleneck for most farms. |
| After scaling out a farm, the timer job does not appear to finish before its next scheduled run. | The Word Automation Services’ timer job’s execution time will scale linearly with the number of Word Automation Services –enabled application servers in the farm. Eventually, it’s possible that the timer job will take longer than one minute to complete its run. | No action is needed if this occurs. SharePoint will simply not begin a scheduled timer job run if the previous run is still executing. |