ThingWorx Platform 8.x Sizing Guide

Version 1.0

February 2019

PTC Inc., 140 Kendrick Street, Needham, MA 02494 USA

Contents

Document Revision History

Revision Date	Version	Description of Change
February 2019	1.0	Initial document version.

About This Guide

The intent of this guide is to provide the reader with a useful method to estimate the amount of processing and memory that a ThingWorx 8 system may need to meet your requirements. Considerations for both on-premise and cloud deployments are provided.

The guidance provided in this document is from the analysis of test data. Many performance load test scenarios were executed against various sized ThingWorx systems. The test results were analyzed to develop the small, medium, and large thresholds discussed below, as well as other tips and guidance.

Note

- This guide should not be used as a benchmark that defines ThingWorx scalability limits. It is intended to provide hardware estimates for ThingWorx instances.
- The guidance provided in this document is intended to support a majority of sizing requests. The guidance numbers used to designate small, medium, or large should not be construed as the ceiling for that system. If your requirements exceed the guidance discussed in this document, please contact PTC to review your use case.
- This Sizing guide does not discuss scaling options for ThingWorx, such as sharding and federation.

1

ThingWorx Hardware Sizing Steps

In general, hardware sizing is driven by the number of things to be managed, their data streaming frequency, and the volume of HTTP requests. Additional sizing considerations are also provided, which depend on your specific use of ThingWorx.

The basic steps for generating a ThingWorx server hardware estimate are listed below. These steps are described in further detail in this guide.

1. Collect ThingWorx usage requirements
2. Calculate key sizing criteria
3. Compare sizing parameter value to guidance
4. Select hardware sizing for on-premise or cloud deployments
5. Consider additional load impacts. These impacts tend to be specific to your deployment

1. Collect ThingWorx Usage Requirements

There are three areas of requirements to collect: HTTP requests, data ingestion, and architecture/deployment.

HTTP Request Requirements

Estimate the mashups and services called during peak usage time.

- Time Period of Peak User Access (t): The length of time where the following mashups and services are called.
- Number of Mashups Called (M): The total number of mashups called during time period (t).
- Number of Services Called (S): The total number of services called during time period (t). This value should include services called by mashups (M).

Data Ingestion Requirements

For each Thing type within a ThingWorx system, estimate the number of Things and the data ingestion rates.

- Number of Things (T): The number of Things (or devices, sensors, connections, modules, etc.) that will be managed by ThingWorx. The number of Things can affect many components of ThingWorx, such as the number of connection servers and platform memory requirements. The number of simultaneous Things connecting to ThingWorx will directly affect the number of connection servers required.
- Properties Per Thing (P): The number of properties (or attributes) that each Thing will send to ThingWorx. Values of these properties are sent from the Thing to ThingWorx at a regular frequency. The number of persisted and logged properties will affect write performance to the database.
- Transmission Frequency (F_D): The frequency of writes from each Thing to ThingWorx. How often will Things submit property values to ThingWorx? This can range from once a week to ten times per second. The number of properties and their write frequency is the major influence on platform size. It is the largest factor in deciding what database solution is needed by ThingWorx to ingest the content.

Common Transmission Rates	Equivalent Daily Transmission Frequency (F_D)
once per day	1
once per hour	24

Common Transmission Rates	Equivalent Daily Transmission Frequency (F_D)
every 15 minutes	96
every 5 minutes	288
every minute	1,440
every 30 seconds	2,880
every second (or 1Hz)	86,400

Architecture/Deployment Requirements

These architecture/deployment requirements may be prudent to consider, depending on your use of them in ThingWorx These considerations have a wide range of impact and are outside the scope of this document.

- File transfers from Things to/from ThingWorx, including software updates.
- Subscriptions, timers, and events generated by your use of ThingWorx.
- The number of normal concurrent tunnel sessions to devices.
- Connections to other services (such as a SCADA, ERP, and other back-office systems.
- Customer data retention policies, especially if the retained content is frequently accessed by ThingWorx.

2. Calculate Key Sizing Criteria

After the requirements are collected, use them as inputs to calculate your key sizing criteria. The calculations mainly shape requirements into common units with PTC's sizing guidelines.

HTTP Requests

For ThingWorx sizing, user access is recognized as the number of concurrent user-driven HTTP requests. We estimate the number of concurrent HTTP requests by summing up the mashup (M) calls and services (S) calls made during a peak usage time (t).

HTTP Requests = (M + S) / t

To match the guidance terms below, the HTTP requests should be on a per-second basis. If the provided time (t) is not in seconds, it should be converted.

Data Ingestion

For each Thing type, calculate the following:

- Number of Things (T) - This value comes directly from the requirements.
- Properties Per Thing (P) - This value comes directory from the requirements.
- Transmission Frequency (F_S) - Convert the given daily frequency (F_D) to a per second rate.

  $F_S = (F_D)$ x (1 day / 24 hours) x (1 hour / 60 minutes) x (1 minute / 60 seconds)
- Operations Per Second (OPS) - The total number of operations that ThingWorx could receive in one second.

  OPS = (T) x (F_S)
- Property Writes Per Second (PWS) - The total number of property writes that ThingWorx may need to write to its database(s).

  PWS = (P) x (OPS)

Take a full count of Things and property writes from all thing types that ThingWorx will manage. Thingcount is the total number of things to be managed, and VS queue is the number of writes coming from the devices to ThingWorx.

Thingcount = ΣT_{types}

VS queue = ΣPWS_{types}

Also, for data ingestion, calculate the number of connection servers (CS) needed to support the connections between devices and ThingWorx. A general rule-of-thumb is to estimate one connection server for every 100,000 connections.

CS = (Thingcount) / 100,000

Additional guidance for sizing connection servers is provided in the ThingWorx Connection Server Installation and Operations Guide.

Your Key Sizing Criteria

Key Sizing Parameter	Definition	Your Value
CS	Number of connection servers to be added	
HTTP Requests	Estimated HTTP page loads on ThingWorx	
Thingcount	Estimated number of Things managed by ThingWorx	
VS Queue	Estimated writes per second to ThingWorx	

3. Compare Sizing Criteria to Guidelines

Compare your sizing criteria against the following guidelines to select an appropriate size rating of small, medium, or large.

ValueStream (VS) Queue Rate

VS Queue is the amount of data that ThingWorx receives and manages from all devices. The max write / sec value here is compared to your VS queue criteria. Choose a size where your VS queue criteria is lower than Max writes / second.

Platform	Max Writes / Second (wps)	Max Writes / Hour (wph)
ThingWorx/ H2 – Extra Small	2,000	7.2 million
ThingWorx / PostgreSQL - Small	17,000	61.2 million
ThingWorx / PostgreSQL - Medium	18,000	64.8 million
ThingWorx / PostgreSQL - Large	21,000	75.6 million
ThingWorx / MS SQL Server - Small	7,000	25.2 million
ThingWorx / MS SQL Server - Medium	10,000	36 million
ThingWorx / MS SQL Server - Large	13,000	46.8 million
ThingWorx Enterprise (PostgreSQL with InfluxDB for ValueStream content))	100,000*	360 million

Note

*ThingWorx and InfluxDB can maintain this ingestion rate when the total number of parameter instances is under 220,000. For example, 10,000 Things, each with 20 properties, need 200,000 parameter instances. If the number of parameter instance is above 220,000, this sizing guide is not able to provide a sizing estimate. ThingWorx and InfluxDB will most likely require fine tuning to achieve the higher ingestion rates with the larger number of parameter instances.

Compare your ValueStream Queue rate against these guidelines to select an appropriately sized system.

Thingcount Comparison

The number of devices connected to ThingWorx. The number of Things managed by ThingWorx has a significant influence on the memory requirements of the platform and has little bearing on CPU utilization. The general guidelines to follow are:

Platform	No. of Devices (or Things)
ThingWorx – Extra Small	10,000
ThingWorx – Small	30,000
ThingWorx – Medium	100,000
ThingWorx – Large	250,000
Contact PTC	Greater than 250,000

Compare your Thingcount against these guidelines to select an appropriately sized system.

HTTP Requests Comparison

The HTTP page loads to be managed on the ThingWorx server. The general guidelines to follow are:

Platform	Max HTTP Requests (Per Sec)	
	PostgreSQL	Microsoft SQL
ThingWorx – Extra Small	10	
ThingWorx – Small	148	1
ThingWorx – Medium	27	4
ThingWorx – Large	2,000	498

Compare your HTTP requests to these guidelines to select an appropriately sized system.

Connection Servers Estimate

Round down the CS value to get a value to the next whole number. For example, 2.3 rounds down to 2. This value will be the suggested number of connections servers to be used in your ThingWorx system.

For each Connection Server, PTC recommends the same hardware specifications that are suggested for a small ThingWorx platform.

4. Select Hardware Sizing for On-Premise or Cloud Deployments

Now, compare the thingcount, value stream, and HTTP request evaluation sizes. The largest size from any of these evaluations should be applied as the overall platform size. In the majority of estimates, PTC expects the Value Stream Queue rate to be the largest factor in determining platform size. The following charts provide comparable AWS, Microsoft Azure, and on-premise specifications for small, medium-sized, and large ThingWorx platforms and databases. With the size now determined, use the charts below to obtain server size metrics.

ThingWorx

Size	AWS EC2	Azure VM	On-Premise CPU Cores	On-Premise Memory (GiB)	Storage Band-width (Mbps)
Extra Small / H2	C4.xlarge	F4v2	4	7.5	750
Small / H2	C4.2xlarge	F8v2	8	15	1,000
Small	C4.2xlarge	F8v2	8	15	1,000
Medium	C4.4xlarge	F16v2	16	30	2,000
Large	C4.8xlarge	F32v2	36	60	4,000
Enterprise	C5.9xlarge	Not Tested	36	72	7,000

PostgreSQL Database

Size	AWS EC2	Azure VM	On-Premise CPU Cores	On-Premise Memory (GiB)	Storage Band-width (Mbps)
Small	C3.2xlarge	F8v2	8	15	1,000
Medium	C3.4xlarge	F16v2	16	30	2,000
Large	C3.8xlarge	F32v2	32	60	4,000

Microsoft SQL (MS SQL) Server Database

Size	AWS EC2	Azure VM	On-Premise CPU Cores	On-Premise Memory (GiB)	Storage Band-width (Mbps)
Small	C3.2xlarge	F8v2	8	15	1,000
Medium	C3.4xlarge	F16v2	16	30	2,000
Large	C3.8xlarge	F32v2	32	60	4,000

InfluxDB Server

Size	AWS EC2	Azure VM	On-Premise CPU Cores	On-Premise Memory (GiB)	Storage Band-width (Mbps)
Enterprise	C3.8xlarge	F32v2	32	60	4,000

Server Terminology

The following content discusses the hardware terminology used in the above charts:

Traditional On-Premise Terminology

Traditional or on-premise hardware sizes are typically discussed in terms of CPU cores for processing power and RAM for memory capability. For example, a small ThingWorx Platform using the H2 database may be sized at 8 CPU cores and 15 GB RAM.

Amazon Web Services (AWS) Terminology

For EC2 instances, AWS provides a wide selection of instance types to fit your use cases. PTC Performance testing is done using the Compute Optimized instance types, primarily C4 instance types for ThingWorx Platform and Connection servers, and C3 and C5d instance types for PostgreSQL and MS SQL Server databases. They are defined here by AWS as "...instances that are optimized for compute-intensive workloads and deliver very cost-effective high performance at a low price per compute ratio."

AWS provides a T-shirt methodology for selecting the size of an EC2 instance in terms of CPU and memory. Typical sizing terms are large, xlarge, 2xlarge, etc. Following the example in the above on-premise terminology, a small ThingWorx Platform using the H2 database may be sized to run on a C4.2xlarge EC2 instance. Other EC2 instance types, such as General Purpose (M) or Memory Intensive (R), can also be considered, but are not covered in this guide.

Microsoft Azure

Azure provides a selection of instance types to fit your use cases. PTC recommends the Compute Optimized instance types, primarily the Fv2-Series. They are defined here by MS Azure as VMs that "...sport a higher CPU to memory ratio. They feature 2 GB RAM and 8 GB of local solid-state drive (SSD) per CPU core, and are optimized for compute-intensive workloads."

Azure provides a packaged method for selecting a VM in terms of CPU cores. Typical sizing terms are F2v2, F4v2, F8v2, etc. where the number represents the number of CPU cores in the VM. Following the example in the above on-premise terminology, a small ThingWorx Platform using the H2 database may be sized to run on a F8v2 VM.

5. Additional Platform Loads to Consider

The added load from other solutions, deployment, and other architecture requirements may be prudent to consider, depending on your use of them. Below are some common architecture decisions and operations that may affect hardware sizing for ThingWorx.

High Availability Requirements for ThingWorx

Most high availability requirements will push a ThingWorx system to incorporate a high availability architecture, such as that described in the ThingWorx HA guide. The added components of a PostgreSQL high availability system and added processing (load balancing, replication, etc) can cause a slight reduction in write performance, enough to require consideration when sizing a ThingWorx system.

File Vaulting / Management

Will any file content (images, pdf files, etc.) be transferred from the Things? In most Remote Service business scenarios, File Upload (from device to platform) is basic a requirement. These files contain anything from log files to images generated by the device (needed in the platform for troubleshooting) or other calibration data. Also, files pushed or downloaded (from platform to device) is another common use case, such as pushing calibration data, software updates, etc.

Subscriptions and Events

Subscriptions, timers, and events can add load; however, this is very specific to your implementation and is out of scope for general coverage provided in this document.

Database Choices

A database choice may have been derived previously from high availability requirements, customer comfort and experience, etc. The database choice, however, does have a role in ThingWorx server sizing.

- H2 is an out-of-the-box database supplied as part of ThingWorx. PTC provides it as a useful database for development and small production systems. The use of H2, however, does not scale well past small implementations.
- PostgreSQL is a supported database to manage the ThingWorx data model, streams, and value streams in development and production systems. It will scale for all small, medium-sized, and large implementations.
- Microsoft SQL Server is a supported database to manage the ThingWorx data model, streams, and value streams in development and production systems. It will scale for all small, medium-sized, and large implementations.
- InfluxDB is a supported database to manage ThingWorx Streams and ValueStreams in development and production systems. It is not supported to manage the ThingWorx model. InfluxDB is used to provide higher ingestion rates.

2

Platform Sizing Examples

This section lists few examples that walk you through the sizing process.

Example 1: Large number of things, small number of properties, low write frequency

Scenario

Monitoring 100,000 water pumps throughout the area. Each water pump reports 20 property values to ThingWorx every five minutes. During peak user access, 10 mashups will each be called 1,000 times in one hour. These mashups each call 10 services every time they are accessed. This ThingWorx configuration use a PostgreSQL database.

Requirements

- Number of Thing Types: 1
- Number of Things: 100,000
- Number of Properties: 20
- Write Frequency: 288 writes per day, per property
- Peak Usage Period: 1 hour
- Number of Mashups Called: (10 mashups) * (1000 calls/mashup) = 10,000 calls/hr
- Number of Services Per Mashup: 10
- PostgreSQL Database

Calculations

- Number of Things (T) = 100,000
- Thingcount = 100,000
- CS = 100,000 / 100,000
- CS = 1
- Number of Properties Per Thing (P) = 20
- Transmission Frequency (F_S) = (288 writes/day) (1 day / 24 hours) (1 hour / 60 minutes) (1 minute / 60 seconds) = 0.0033 write / sec
- Operations Per Second (OPS) = (100,000 things) (0.003 write / sec) = 330 ops
- Property Writes Per Second (PWS) = (330 ops) (20 properties) = 6,600 wps
- VS Queue Rate = 6,600 wps
- Mashup Calls / Sec = (10,000 calls / hr) / (60 min / hr) / (60 sec / min) = 2.78 mashup calls / sec
- Service Calls / Sec = (2.78 mashup calls / sec) * (10 services calls / mashup) = 27.8 service calls / sec

- HTTP Requests = Mashups calls / sec + Service calls / sec = 2.78 + 27.8
- HTTP Requests = 31 calls / sec

Criteria Comparison

- Thingcount = 100,000. This estimate is larger than a small thingcount of 30,000, and equal a medium thingcount of 100,000. A medium-sized ThingWorx Platform (with PostgreSQL) is sufficient.
- CS = 1. One connection server is recommended.
- VS Queue rate = 6,600. This estimate is lower than the small max queue rate of 7,000 wps. A small ThingWorx Platform (with PostgreSQL) is sufficient.
- HTTP Request = 31. A small ThingWorx Platform is sufficient.

Sizing

Reviewing all estimates, a medium-sized ThingWorx system will satisfy all criteria. Reviewing the above charts, a medium-sized system is:

ThingWorx

Size	AWS EC2 Instance	Azure VM	On-Premise CPU Cores	On-Premise Memory (GiB)
Medium	C4.4xlarge	F16v2	16	30

PostgreSQL Database

Size	AWS EC2 Instance	Azure VM	On-Premise CPU Cores	On-Premise Memory (GiB)
Medium	C3.4xlarge	F16v2	16	30

ThingWorx Connection Server

Size	AWS EC2 Instance	Azure VM	On-Premise CPU Cores	On-Premise Memory (GiB)	Quantity
	C4.2xlarge	F8v2	8	15	1

Example 2: Small number of things, small number of properties, high write frequency

Scenario

A medium-sized factory where 250 machines are monitored. Each monitored machine is sending 50 property results to ThingWorx every second. During their peak usage hour, there are 100 users that make 100 mashup calls, with mashups averaging 10 services call per request. This ThingWorx configuration uses a Microsoft SQL Server database.

Requirements

- Number of Thing Types: 1
- Number of Things: 250
- Number of Properties: 50
- Write Frequency: 86,400 writes per day, per property
- Peak User Access Period = 1 hour
- Number of Users: 100
- Mashups Called Per User = 100
- Average Number of Services Per Mashup = 10

Calculations

- Number of Things (T) = 250
- Thingcount = 250
- CS = round down (250 / 50,000)
- CS = 0
- Number of Properties Per Thing (P) = 50
- Transmission Frequency (F_S) = (86,400 writes/day) (1 day / 24 hours) (1 hour / 60 minutes) (1 minute / 60 seconds) = 1 write / sec
- Operations Per Second (OPS) = (250 things) (1 write / sec) = 250 ops
- Property Writes Per Second (PWS) = (250 ops) (50 properties) = 12,500 wps
- VS Queue Rate = 12,500 wps
- Time Period = 1 hour = 3,600 sec
- Mashup Calls = 100 users x 100 mashup requests = 10,000 requests / hour = 2.78 requests / sec

- Service Calls = 2.78 requests / sec x 10 service calls / request = 27.8 service calls / sec
- HTTP Requests = 2.78 mashup requests / sec + 27.8 services calls / sec
- HTTP Requests = 31

Criteria Comparison

- Thingcount = 250. This estimate is smaller than a small thingcount of 30,000. A small ThingWorx Platform is sufficient.
- VS Queue Rate = 12,500. This estimate is higher than the small queue rate of 7,000 wps and lower than the medium queue rate of 17,000 wps. A medium-sized ThingWorx Platform (with MS SQL Server) is sufficient.
- HTTP Request = 31. A small ThingWorx Platform is sufficient.
- CS = 0. No connection servers are necessary.

Sizing

Comparing all criteria, a medium-sized ThingWorx system will satisfy all criteria. Reviewing the above charts, a medium-sized system is:

ThingWorx

Size	AWS EC2 Instance	Azure VM	On-Premise CPU Cores	On-Premise Memory (GiB)
Medium	C4.4xlarge	F16v2	16	30

MS SQL Server Database

Size	AWS EC2 Instance	Azure VM	On-Premise CPU Cores	On-Premise Memory (GiB)
Medium	C3.4xlarge	F16v2	16	30

A

PTC Test Run Summaries

This section describes the test runs used to develop this guide. For configuration and tuning, the ThingWorx Install guide was followed. No other configuration or tuning actions were applied.

Extra Small Servers Test (using H2)

A test of ThingWorx performance with H2 using 4 CPU cores and 7.5 GB RAM.

Hardware Configuration

AWS EC2 Instance Type	C3.xlarge
vCPU	4
Memory	7.5 GB
Storage	80 GB (SSD)

Test Scenario

Basic Configuration			
Number of Things	10,000		
Number of Templates	40		
Number of Properties	20		
Property Types	Integer 10	String 10	
Number of Services	20		
Properties with Alerts	50%		
Alerts with Subscrip-tions	50%		
Things with…	Percent	Number of Things	Number of Properties
Simple Properties	20%	2,000	40,000
Logged Properties	68%	6,800	136,000
Persistent Properties	2%	200	4,000
Read-Only Properties	10%	1,000	20,000
Total Number of	100%	10,000	200,000

Basic Configuration					
Things					
Write Operations	Percent	Number of Things			
Chatty	20%	1,800			
Non-Chatty	80%	7,200			
Configuration for Streams					
Number of Streams	20				
Number of Data Shapes	2				
Property Types Per Template	Integer, String				
Number of Columns	10				
Data Tables					
Table Type	Number Tables	Data Shapes	Initial Rows	Property Types	Fields / Type
Large Tables	10	2	1,000	Integer, String	10
Lookup Tables	25	2	10	String	1
Configuration for External Subscriptions					
Number of Alerting Things	20				
Number of Subscrip-tions	2				
Mashups / Read					
Operation	Total Users	Max Items			
Mashup (property, value stream)	500	100			

Basic Configuration		
Mashup (stream)	250	100
Mashup (data tables)	500	100
Users		
Administrators	100	
Non-Administrators	1,000	

Test Results

Test Results Summary	
CPU Utilization	70.22% of 2 CPU cores
Memory Utilization	6.73 of 7.3 GB (92.19%)
Websocket Requests (writes)	249 wps
HTTP Requests (reads)	10 rps
Value Stream Queue Rate	2,000 wps
Stream Queue Rate	1 wps
Alerts Queue Rate	0 ops
Events Queue Rate	0 ops

Platform Subsystem- GetPerformanceMetrics		
Name	**Description**	**Value**
eventQueueSize	Event queue size	0
streamQueueSize-ThingworxPersistence-Provider	Stream queue size	0
valueStreamQueueSize-ThingworxPersistence-Provider	Value Stream queue size	0
memoryInUse	Memory in use (bytes)	2,738,006,216
totalMemoryAllocated	Total memory allocated (bytes)	5,882,511,360
thingCount	Thing count	15,607

Value Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
maximumWaitTime	ThingworxPersistence-Provider: Maximum wait time before flushing stream buffer (milliseconds)	10,000
sizeThreshold	ThingworxPersistence-Provider: Maximum number of items accumulated before flushing stream buffer	1,000
maximumBlockSize	ThingworxPersistence-Provider: Maximum number of stream writes processed in one block	2,500
scanRate	ThingworxPersistence-Provider: Rate stream queue is checked (milliseconds)	5
maximumQueueSize	ThingworxPersistence-Provider: Maximum number of stream entries to queue	2,000,000
queueSize	ThingworxPersistence-Provider: Number of stream entries currently queued	0
totalWritesQueued	ThingworxPersistence-Provider: Number of stream entries that have been queued	14,858,151
totalWritesPerformed	ThingworxPersistence-Provider: Number of stream entries that have been performed	11,947,408
numberOfProcessingThreads	ThingworxPersistence-Provider: Number of processing threads	10

Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
maximumWaitTime	ThingworxPersistence-Provider: Maximum wait time before flushing stream buffer (milliseconds)	10,000
sizeThreshold	ThingworxPersistence-Provider: Maximum number of items accumulated before flushing stream buffer	1,000
maximumBlockSize	ThingworxPersistence-Provider: Maximum number of stream writes processed in one block	2,500
scanRate	ThingworxPersistence-Provider: Rate stream queue is checked (milliseconds)	5
maximumQueueSize	ThingworxPersistence-Provider: Maximum number of stream entries to queue	1,000,000
queueSize	ThingworxPersistence-Provider: Number of stream entries currently queued	0
totalWritesQueued	ThingworxPersistence-Provider: Number of stream entries that have been queued	10,760
totalWritesPerformed	ThingworxPersistence-Provider: Number of stream entries that have been performed	107,608,260
numberOfProcessingThreads	ThingworxPersistence-Provider: Number of processing threads	10

Small Servers Test (using PostgreSQL)

A test of ThingWorx performance with PostgreSQL using two servers with 8 CPU cores and 15 GB RAM.

Hardware Configuration

Server Purpose	ThingWorx	PostgreSQL
AWS EC2 Instance Type	C4.2xlarge	C3.2xlarge
vCPU	8	8
Memory	15 GB	15 GB
Storage	1,000 Mbps	160 GB (SSD)

Test Scenario

Basic Configuration			
Number of Things	30,000		
Number of Templates	40		
Number of Properties	20		
Property Types	Integer 10	String 10	
Number of Services	20		
Properties with Alerts	50%		
Alerts with Subscrip-tions	50%		
Things with…	Percent	Number of Things	Number of Properties
Simple Properties	20%	6,000	120,000
Logged Properties	68%	20,400	408,000
Persistent Properties	2%	600	12,000
Read-Only Properties	10%	3,000	60,000

Basic Configuration					
Total Number of Things	100%	30,000	600,000		
Write Operations	Percent	Number of Things			
Chatty	20%	5,400			
Non-Chatty	80%	21,600			
Configuration for Streams					
Number of Streams	20				
Number of Data Shapes	2				
Property Types Per Template	Integer, String				
Number of Columns	10				
Data Tables					
Table Type	Number Tables	Data Shapes	Initial Rows	Property Types	Fields/Type
Large Tables	10	2	1,000	Integer, String	10
Lookup Tables	25	2	10	String	1
Configuration for External Subscriptions					
Number of Alerting Things	20				
Number of Subscrip-tions	2				
Mashups / Read					
Operation	Total Users	Max Items			
Mashup (property,	500	100			

Basic Configuration		
value stream)		
Mashup (stream)	250	100
Mashup (data tables)	500	100
Users		
Administrators	100	
Non-Administrators	1,000	

Test Results

Test Results Summary		
Server	**ThingWorx**	**PostgreSQL**
CPU Utilization	56.61% of 8 CPU cores	53.83% of 8 CPU cores
Memory Utilization	12.54 of 14.7 GB (85.36%)	1.67 of 14.7 GB (11.36%)
Websocket Requests (writes)	2,372 wps	
HTTP Requests (reads)	148 rps	
Value Stream Queue Rate	18,000 wps	
Stream Queue Rate	2 wps	
Alerts Queue Rate	0 ops	
Events Queue Rate	0 ops	

Platform Subsystem- GetPerformanceMetrics		
Name	**Description**	**Value**
eventQueueSize	Event queue size	0
streamQueueSize-ThingworxPersistence-Provider	Stream queue size	0
valueStreamQueueSize-ThingworxPersistence-Provider	Value Stream queue size	0
memoryInUse	Memory in use (bytes)	7,404,122,056

Platform Subsystem- GetPerformanceMetrics		
Name	**Description**	**Value**
totalMemoryAllocated	Total memory allocated (bytes)	11,827,937,280
thingCount	Thing count	31,190

Value Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
maximumWaitTime	ThingworxPersistence-Provider: Maximum wait time before flushing stream buffer (milliseconds)	10,000
sizeThreshold	ThingworxPersistence-Provider: Maximum number of items accumulated before flushing stream buffer	1,000
maximumBlockSize	ThingworxPersistence-Provider: Maximum number of stream writes processed in one block	2,500
scanRate	ThingworxPersistence-Provider: Rate stream queue is checked (milliseconds)	5
maximumQueueSize	ThingworxPersistence-Provider: Maximum number of stream entries to queue	4,000,000
queueSize	ThingworxPersistence-Provider: Number of stream entries currently queued	0
totalWritesQueued	ThingworxPersistence-Provider: Number of stream entries that have been queued	94,551,661

Value Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
totalWritesPerformed	ThingworxPersistence-Provider: Number of stream entries that have been performed	92,734,923
numberOfProcessingThreads	ThingworxPersistence-Provider: Number of processing threads	50

Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
maximumWaitTime	ThingworxPersistence-Provider: Maximum wait time before flushing stream buffer (milliseconds)	10,000
sizeThreshold	ThingworxPersistence-Provider: Maximum number of items accumulated before flushing stream buffer	1,000
maximumBlockSize	ThingworxPersistence-Provider: Maximum number of stream writes processed in one block	2,500
scanRate	ThingworxPersistence-Provider: Rate stream queue is checked (milliseconds)	5
maximumQueueSize	ThingworxPersistence-Provider: Maximum number of stream entries to queue	2,000,000
queueSize	ThingworxPersistence-Provider: Number of stream entries currently queued	0
totalWritesQueued	ThingworxPersistence-Provider: Number of stream entries that have	25,880

Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
	been queued	
totalWritesPerformed	ThingworxPersistence-Provider: Number of stream entries that have been performed	25,873
numberOfProcessingTh-reads	ThingworxPersistence-Provider: Number of processing threads	50

Medium-Sized Servers Test (using PostgreSQL)

A test of ThingWorx performance with PostgreSQL using two servers with 16 CPU cores and 30 GB RAM.

Hardware Configuration

Server Purpose	ThingWorx	PostgreSQL
AWS EC2 Instance Type	C4.4xlarge	C3.4xlarge
vCPU	16	16
Memory	30 GB	30 GB
Storage	2,000 Mbps	320 GB (SSD)

Test Scenario

Basic Configuration		
Number of Things	100,000	
Number of Templates	40	
Number of Properties	20	
Property Types	Integer 10	String 10
Number of Services	20	
Properties with Alerts	50%	
Alerts with	50%	

Basic Configuration					
Subscrip-tions					
Things with…	Percent	Number of Things	Number of Properties		
Simple Properties	20%	20,000	400,000		
Logged Properties	68%	68,000	1,360,000		
Persistent Properties	2%	2,000	40,000		
Read-Only Properties	10%	10,000	200,000		
Total Number of Things	100%	100,000	2,000,000		
Write Operations	Percent	Number of Things			
Chatty	20%	18,000			
Non-Chatty	80%	72,000			
Configuration for Streams					
Number of Streams	20				
Number of Data Shapes	2				
Property Types Per Template	Integer, String				
Number of Columns	10				
Data Tables					
Table Type	Number Tables	Data Shapes	Initial Rows	Property Types	Fields / Type
Large Tables	10	2	1,000	Integer, String	10
Lookup Tables	25	2	10	String	1

Basic Configuration		
Configuration for External Subscriptions		
Number of Alerting Things	20	
Number of Subscrip-tions	2	
Mashups / Read		
Operation	Total Users	Max Items
Mashup (property, value stream)	500	100
Mashup (stream)	250	100
Mashup (data tables)	500	100
Users		
Administra-tors	100	
Non-Administra-tors	1,000	

Test Results

Test Results Summary		
Server	**ThingWorx**	**PostgreSQL**
CPU Utilization	36.20% of 16 CPU cores	53.10% of 16 CPU cores
Memory Utilization	12.55 of 29.4 GB (21.28%)	2.90 of 29.4 GB (4.91%)
Websocket Requests (writes)	2,318 wps	
HTTP Requests (reads)	27 rps	
Value Stream Queue Rate	18,000 wps	
Stream Queue Rate	3 wps	
Alerts Queue Rate	0 ops	
Events Queue Rate	0 ops	

Platform Subsystem- GetPerformanceMetrics		
Name	**Description**	**Value**
eventQueueSize	Event queue size	0
streamQueueSize-ThingworxPersistence-Provider	Stream queue size	0
valueStreamQueueSize-ThingworxPersistence-Provider	Value Stream queue size	0
memoryInUse	Memory in use (bytes)	5,443,677,312
totalMemoryAllocated	Total memory allocated (bytes)	23,710,400,512
thingCount	Thing count	104,367

Value Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
maximumWaitTime	ThingworxPersistence-Provider: Maximum wait time before flushing stream buffer (milliseconds)	10,000
sizeThreshold	ThingworxPersistence-Provider: Maximum number of items accumulated before flushing stream buffer	1,000
maximumBlockSize	ThingworxPersistence-Provider: Maximum number of stream writes processed in one block	2,500
scanRate	ThingworxPersistence-Provider: Rate stream queue is checked (milliseconds)	5
maximumQueueSize	ThingworxPersistence-Provider: Maximum number of stream entries to queue	4,000,000
queueSize	ThingworxPersistence-Provider: Number of	0

Value Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
	stream entries currently queued	
totalWritesQueued	ThingworxPersistence-Provider: Number of stream entries that have been queued	117,113,905
totalWritesPerformed	ThingworxPersistence-Provider: Number of stream entries that have been performed	101,382,136
numberOfProcessingThreads	ThingworxPersistence-Provider: Number of processing threads	50

Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
maximumWaitTime	ThingworxPersistence-Provider: Maximum wait time before flushing stream buffer (milliseconds)	10,000
sizeThreshold	ThingworxPersistence-Provider: Maximum number of items accumulated before flushing stream buffer	1,000
maximumBlockSize	ThingworxPersistence-Provider: Maximum number of stream writes processed in one block	2,500
scanRate	ThingworxPersistence-Provider: Rate stream queue is checked (milliseconds)	5
maximumQueueSize	ThingworxPersistence-Provider: Maximum number of stream entries to queue	2,000,000

Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
queueSize	ThingworxPersistence-Provider: Number of stream entries currently queued	0
totalWritesQueued	ThingworxPersistence-Provider: Number of stream entries that have been queued	53,040
totalWritesPerformed	ThingworxPersistence-Provider: Number of stream entries that have been performed	53,040
numberOfProcessingThreads	ThingworxPersistence-Provider: Number of processing threads	50

Large Servers Test (using PostgreSQL)

A test of ThingWorx performance with PostgreSQL using two servers with 36 CPU cores and 60 GB RAM.

Hardware Configuration

Server Purpose	ThingWorx	PostgreSQL
AWS EC2 Instance Type	C4.8xlarge	C3.8xlarge
vCPU	36	32
Memory	60 GB	60 GB
Storage	4,000 Mbps	640 GB (SSD)

Test Scenario

Basic Configuration		
Number of Things	250,000	
Number of Templates	40	
Number of Properties	20	
Property Types	Integer 10	String 10

Basic Configuration					
Number of Services	20				
Properties with Alerts	50%				
Alerts with Subscrip-tions	50%				
Things with…	Percent	Number of Things	Number of Properties		
Simple Properties	20%	50,000	1,000,000		
Logged Properties	68%	170,000	3,400,000		
Persistent Properties	2%	5,000	100,000		
Read-Only Properties	10%	25,000	500,000		
Total Number of Things	100%	250,000	5,000,000		
Write Operations	Percent	Number of Things			
Chatty	20%	45,000			
Non-Chatty	80%	180,000			
Configuration for Streams					
Number of Streams	20				
Number of Data Shapes	2				
Property Types Per Template	Integer, String				
Number of Columns	10				
Data Tables					
Table Type	Number	Data Shapes	Initial Rows	Property	Fields /

Basic Configuration					
	Tables			Types	Type
Large Tables	10	2	1,000	Integer, String	10
Lookup Tables	25	2	10	String	1
Configuration for External Subscriptions					
Number of Alerting Things	20				
Number of Subscrip-tions	2				
Mashups / Read					
Operation	Total Users	Max Items			
Mashup (property, value stream)	1,000	100			
Mashup (stream)	500	100			
Mashup (data tables)	1,000	100			
Users					
Administra-tors	100				
Non-Administra-tors	3,000				

Test Results

Test Results Summary		
Server	**ThingWorx**	**PostgreSQL**
CPU Utilization	62.69% of 32 CPU cores	48.46% of 32 CPU cores
Memory Utilization	48.6 of 59.0 GB (82.41%)	4.32 of 59.0 GB (7.32%)
Websocket Requests (writes)	5,300 wps	
HTTP Requests (reads)	2,000 rps	

Test Results Summary		
Server	**ThingWorx**	**PostgreSQL**
Value Stream Queue Rate	21,000 wps	
Stream Queue Rate	8 wps	
Alerts Queue Rate	0 ops	
Events Queue Rate	0 ops	

Platform Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
eventQueueSize	Event queue size	0
streamQueueSize - ThingworxPersistence-Provider	Stream queue size	0
valueStreamQueueSize - ThingworxPersistence-Provider	Value Stream queue size	0
memoryInUse	Memory in use (bytes)	17,540,487,456
totalMemoryAllocated	Total memory allocated (bytes)	47,496,298,496
thingCount	Thing count	270,804

Value Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
maximumWaitTime	ThingworxPersistence-Provider: Maximum wait time before flushing stream buffer (milliseconds)	10,000
sizeThreshold	ThingworxPersistence-Provider: Maximum number of items accumulated before flushing stream buffer	1,000
maximumBlockSize	ThingworxPersistence-Provider: Maximum number of stream writes processed in one block	2,500
scanRate	ThingworxPersistence-Provider: Rate stream queue is checked	5

Value Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
	(milliseconds)	
maximumQueueSize	ThingworxPersistence-Provider: Maximum number of stream entries to queue	4,000,000
queueSize	ThingworxPersistence-Provider: Number of stream entries currently queued	0
totalWritesQueued	ThingworxPersistence-Provider: Number of stream entries that have been queued	84,179,260
totalWritesPerformed	ThingworxPersistence-Provider: Number of stream entries that have been performed	60,904,163
numberOfProcessingThreads	ThingworxPersistence-Provider: Number of processing threads	50

Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
maximumWaitTime	ThingworxPersistence-Provider: Maximum wait time before flushing stream buffer (milliseconds)	10,000
sizeThreshold	ThingworxPersistence-Provider: Maximum number of items accumulated before flushing stream buffer	1,000
maximumBlockSize	ThingworxPersistence-Provider: Maximum number of stream writes processed in one block	2,500
scanRate	ThingworxPersistence-Provider: Rate stream	5

Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
	queue is checked (milliseconds)	
maximumQueueSize	ThingworxPersistence-Provider: Maximum number of stream entries to queue	2,000,000
queueSize	ThingworxPersistence-Provider: Number of stream entries currently queued	0
totalWritesQueued	ThingworxPersistence-Provider: Number of stream entries that have been queued	113,040
totalWritesPerformed	ThingworxPersistence-Provider: Number of stream entries that have been performed	113,040
numberOfProcessingThreads	ThingworxPersistence-Provider: Number of processing threads	50

Small Servers Test (using MS SQL)

A test of ThingWorx performance with MS SQL using two servers with 8 CPU cores and 15 GB RAM.

Hardware Configuration

Server Purpose	ThingWorx	MS SQL
AWS EC2 Instance Type	C4.2xlarge	C3.2xlarge
vCPU	8	8
Memory	15 GB	15 GB
Storage	1,000 Mbps	160 GB (SSD)

Test Scenario

Basic Configuration			
Number of Things	30,000		
Number of Templates	40		
Number of Properties	20		
Property Types	Integer 10	String 10	
Number of Services	20		
Properties with Alerts	50%		
Alerts with Subscrip-tions	50%		
Things with…	Percent	Number of Things	Number of Properties
Simple Properties	20%	6,000	120,000
Logged Properties	68%	20,400	408,000
Persistent Properties	2%	600	12,000
Read-Only Properties	10%	3,000	60,000
Total Number of Things	100%	30,000	600,000
Write Operations	Percent	Number of Things	
Chatty	20%	5,400	
Non-Chatty	80%	21,600	
Configuration for Streams			
Number of Streams	20		

Basic Configuration					
Number of Data Shapes	2				
Property Types Per Template	Integer, String				
Number of Columns	10				
Data Tables					
Table Type	Number Tables	Data Shapes	Initial Rows	Property Types	Fields / Type
Large Tables	10	2	1,000	Integer, String	10
Lookup Tables	25	2	10	String	1
Configuration for External Subscriptions					
Number of Alerting Things	20				
Number of Subscrip-tions	2				
Mashups / Read					
Operation	Total Users	Max Items			
Mashup (property, value stream)	500	100			
Mashup (stream)	250	100			
Mashup (data tables)	500	100			
Users					

Basic Configuration	
Administrators	100
Non-Administrators	1,000

Test Results

Test Results Summary		
Server	**ThingWorx**	**MS SQL**
CPU Utilization	21.37% of 8 CPU cores	36.22% of 8 CPU cores
Memory Utilization	12.46 of 14.7 GB (84.81%)	12.99 of 14.7 GB (88.42%)
Websocket Requests (writes)	1,370 wps	
HTTP Requests (reads)	1 rps	
Value Stream Queue Rate	7,000 wps	
Stream Queue Rate	1 wps	
Alerts Queue Rate	34 ops	
Events Queue Rate	253 ops	

Platform Subsystem- GetPerformanceMetrics		
Name	**Description**	**Value**
eventQueueSize	Event queue size	0
streamQueueSize-ThingworxPersistence-Provider	Stream queue size	0
valueStreamQueueSize-ThingworxPersistence-Provider	Value Stream queue size	0
memoryInUse	Memory in use (bytes)	7,404,122,056
totalMemoryAllocated	Total memory allocated (bytes)	11,827,937,280
thingCount	Thing count	31,190

Value Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
maximumWaitTime	ThingworxPersistence-Provider: Maximum wait time before flushing stream buffer (milliseconds)	10,000
sizeThreshold	ThingworxPersistence-Provider: Maximum number of items accumulated before flushing stream buffer	1,000
maximumBlockSize	ThingworxPersistence-Provider: Maximum number of stream writes processed in one block	2,500
scanRate	ThingworxPersistence-Provider: Rate stream queue is checked (milliseconds)	5
maximumQueueSize	ThingworxPersistence-Provider: Maximum number of stream entries to queue	4,000,000
queueSize	ThingworxPersistence-Provider: Number of stream entries currently queued	0
totalWritesQueued	ThingworxPersistence-Provider: Number of stream entries that have been queued	94,551,661
totalWritesPerformed	ThingworxPersistence-Provider: Number of stream entries that have been performed	92,734,923
numberOfProcessingThreads	ThingworxPersistence-Provider: Number of processing threads	50

Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
maximumWaitTime	ThingworxPersistence-Provider: Maximum wait time before flushing stream buffer (milliseconds)	10,000
sizeThreshold	ThingworxPersistence-Provider: Maximum number of items accumulated before flushing stream buffer	1,000
maximumBlockSize	ThingworxPersistence-Provider: Maximum number of stream writes processed in one block	2,500
scanRate	ThingworxPersistence-Provider: Rate stream queue is checked (milliseconds)	5
maximumQueueSize	ThingworxPersistence-Provider: Maximum number of stream entries to queue	2,000,000
queueSize	ThingworxPersistence-Provider: Number of stream entries currently queued	0
totalWritesQueued	ThingworxPersistence-Provider: Number of stream entries that have been queued	25,880
totalWritesPerformed	ThingworxPersistence-Provider: Number of stream entries that have been performed	25,873
numberOfProcessingThreads	ThingworxPersistence-Provider: Number of processing threads	50

Medium-Sized Servers Test (using MS SQL)

A test of ThingWorx performance with MS SQL using two servers with 16 CPU cores and 30 GB RAM.

Hardware Configuration

Server Purpose	ThingWorx	MS SQL
AWS EC2 Instance Type	C4.4xlarge	C3.4xlarge
vCPU	16	16
Memory	30 GB	30 GB
Storage	2,000 Mbps	320 GB (SSD)

Test Scenario

Basic Configuration			
Number of Things	100,000		
Number of Templates	40		
Number of Properties	20		
Property Types	Integer 10	String 10	
Number of Services	20		
Properties with Alerts	50%		
Alerts with Subscrip-tions	50%		
Things with…	Percent	Number of Things	Number of Properties
Simple Properties	20%	20,000	400,000
Logged Properties	68%	68,000	1,360,000
Persistent Properties	2%	2,000	40,000

Basic Configuration					
Read-Only Properties	10%	10,000	200,000		
Total Number of Things	100%	100,000	2,000,000		
Write Operations	Percent	Number of Things			
Chatty	20%	18,000			
Non-Chatty	80%	72,000			
Configuration for Streams					
Number of Streams	20				
Number of Data Shapes	2				
Property Types Per Template	Integer, String				
Number of Columns	10				
Data Tables					
Table Type	Number Tables	Data Shapes	Initial Rows	Property Types	Fields / Type
Large Tables	10	2	1,000	Integer, String	10
Lookup Tables	25	2	10	String	1
Configuration for External Subscriptions					
Number of Alerting Things	20				
Number of Subscrip-tions	2				
Mashups / Read					
Operation	Total Users	Max Items			

Basic Configuration		
Mashup (property, value stream)	500	100
Mashup (stream)	250	100
Mashup (data tables)	500	100
Users		
Administrators	100	
Non-Administrators	1,000	

Test Results

Test Results Summary		
Server	**ThingWorx**	**MS SQL**
CPU Utilization	17.43% of 16 CPU cores	16.99% of 16 CPU cores
Memory Utilization	9.97 of 29.4 GB (33.86%)	4.47 of 29.4 GB (15.18%)
Websocket Requests (writes)	1,640 wps	
HTTP Requests (reads)	4 rps	
Value Stream Queue Rate	10,000 wps	
Stream Queue Rate	2 wps	
Alerts Queue Rate	34 ops	
Events Queue Rate	253 ops	

Platform Subsystem- GetPerformanceMetrics		
Name	**Description**	**Value**
eventQueueSize	Event queue size	0
streamQueueSize-ThingworxPersistence-Provider	Stream queue size	0
valueStreamQueueSize-ThingworxPersistence-Provider	Value Stream queue size	0
memoryInUse	Memory in use (bytes)	14,443,028,872

Platform Subsystem- GetPerformanceMetrics		
Name	**Description**	**Value**
totalMemoryAllocated	Total memory allocated (bytes)	22,158,508,032
thingCount	Thing count	104,361

Value Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
maximumWaitTime	ThingworxPersistence-Provider: Maximum wait time before flushing stream buffer (milliseconds)	10,000
sizeThreshold	ThingworxPersistence-Provider: Maximum number of items accumulated before flushing stream buffer	1,000
maximumBlockSize	ThingworxPersistence-Provider: Maximum number of stream writes processed in one block	2,500
scanRate	ThingworxPersistence-Provider: Rate stream queue is checked (milliseconds)	5
maximumQueueSize	ThingworxPersistence-Provider: Maximum number of stream entries to queue	4,000,000
queueSize	ThingworxPersistence-Provider: Number of stream entries currently queued	0
totalWritesQueued	ThingworxPersistence-Provider: Number of stream entries that have been queued	40,954,390

Value Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
totalWritesPerformed	ThingworxPersistence-Provider: Number of stream entries that have been performed	16,852,590
numberOfProcessingThreads	ThingworxPersistence-Provider: Number of processing threads	50

Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
maximumWaitTime	ThingworxPersistence-Provider: Maximum wait time before flushing stream buffer (milliseconds)	10,000
sizeThreshold	ThingworxPersistence-Provider: Maximum number of items accumulated before flushing stream buffer	1,000
maximumBlockSize	ThingworxPersistence-Provider: Maximum number of stream writes processed in one block	2,500
scanRate	ThingworxPersistence-Provider: Rate stream queue is checked (milliseconds)	5
maximumQueueSize	ThingworxPersistence-Provider: Maximum number of stream entries to queue	2,000,000
queueSize	ThingworxPersistence-Provider: Number of stream entries currently queued	0
totalWritesQueued	ThingworxPersistence-Provider: Number of stream entries that have	29,230

Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
	been queued	
totalWritesPerformed	ThingworxPersistence-Provider: Number of stream entries that have been performed	29,166
numberOfProcessingThreads	ThingworxPersistence-Provider: Number of processing threads	50

Large Servers Test (using MS SQL)

A test of ThingWorx performance with MS SQL using two servers with 36 CPU cores and 60 GB RAM.

Hardware Configuration

Server Purpose	ThingWorx	MS SQL
AWS EC2 Instance Type	C4.8xlarge	C3.8xlarge
vCPU	36	32
Memory	60 GB	60 GB
Storage	4,000 Mbps	640 GB (SSD)

Test Scenario

Basic Configuration		
Number of Things	250,000	
Number of Templates	40	
Number of Properties	20	
Property Types	Integer 10	String 10
Number of Services	20	
Properties with Alerts	50%	
Alerts with Subscriptions	50%	

Basic Configuration					
Things with…	Percent	Number of Things	Number of Properties		
Simple Properties	20%	50,000	1,000,000		
Logged Properties	68%	170,000	3,400,000		
Persistent Properties	2%	5,000	100,000		
Read-Only Properties	10%	25,000	500,000		
Total Number of Things	100%	250,000	5,000,000		
Write Operations	Percent	Number of Things			
Chatty	20%	45,000			
Non-Chatty	80%	180,000			
Configuration for Streams					
Number of Streams	20				
Number of Data Shapes	2				
Property Types Per Template	Integer, String				
Number of Columns	10				
Data Tables					
Table Type	Number Tables	Data Shapes	Initial Rows	Property Types	Fields / Type
Large Tables	10	2	1,000	Integer, String	10
Lookup Tables	25	2	10	String	1
Configuration for External Subscriptions					

Basic Configuration		
Number of Alerting Things	20	
Number of Subscrip-tions	2	
Mashups / Read		
Operation	Total Users	Max Items
Mashup (property, value stream)	1,000	100
Mashup (stream)	500	100
Mashup (data tables)	1,000	100
Users		
Administra-tors	100	
Non-Administra-tors	3,000	

Test Results

Test Results Summary		
Server	**ThingWorx**	**MS SQL**
CPU Utilization	33.31% of 32 CPU cores	23.24% of 32 CPU cores
Memory Utilization	44.46 of 59.0 GB (75.39%)	10.61 of 59.0 GB (7.99%)
Websocket Requests (writes)	2,046 wps	
HTTP Requests (reads)	498 rps	
Value Stream Queue Rate	13,000 wps	
Stream Queue Rate	6 wps	
Alerts Queue Rate	0 ops	
Events Queue Rate	0 ops	

Platform Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
eventQueueSize	Event queue size	0
streamQueueSize - ThingworxPersistence-Provider	Stream queue size	0
valueStreamQueueSize - ThingworxPersistence-Provider	Value Stream queue size	0
memoryInUse	Memory in use (bytes)	36,937,087,224
totalMemoryAllocated	Total memory allocated (bytes)	47,496,298,496
thingCount	Thing count	270,804

Value Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
maximumWaitTime	ThingworxPersistence-Provider: Maximum wait time before flushing stream buffer (milliseconds)	10,000
sizeThreshold	ThingworxPersistence-Provider: Maximum number of items accumulated before flushing stream buffer	1,000
maximumBlockSize	ThingworxPersistence-Provider: Maximum number of stream writes processed in one block	2,500
scanRate	ThingworxPersistence-Provider: Rate stream queue is checked (milliseconds)	5
maximumQueueSize	ThingworxPersistence-Provider: Maximum number of stream entries to queue	4,000,000
queueSize	ThingworxPersistence-Provider: Number of	0

Value Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
	stream entries currently queued	
totalWritesQueued	ThingworxPersistence-Provider: Number of stream entries that have been queued	84,176,505
totalWritesPerformed	ThingworxPersistence-Provider: Number of stream entries that have been performed	59,496,940
numberOfProcessingThreads	ThingworxPersistence-Provider: Number of processing threads	50

Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
maximumWaitTime	ThingworxPersistence-Provider: Maximum wait time before flushing stream buffer (milliseconds)	10,000
sizeThreshold	ThingworxPersistence-Provider: Maximum number of items accumulated before flushing stream buffer	1,000
maximumBlockSize	ThingworxPersistence-Provider: Maximum number of stream writes processed in one block	2,500
scanRate	ThingworxPersistence-Provider: Rate stream queue is checked (milliseconds)	5
maximumQueueSize	ThingworxPersistence-Provider: Maximum number of stream entries to queue	2,000,000

Stream Subsystem - GetPerformanceMetrics		
Name	**Description**	**Value**
queueSize	ThingworxPersistence-Provider: Number of stream entries currently queued	0
totalWritesQueued	ThingworxPersistence-Provider: Number of stream entries that have been queued	113,040
totalWritesPerformed	ThingworxPersistence-Provider: Number of stream entries that have been performed	113,040
numberOfProcessingThreads	ThingworxPersistence-Provider: Number of processing threads	50