WHY SIZE MATTERS WITH THINGWORX

David Bernbeck
Tech Fellow, Enterprise Deployment Center
AGENDA

1. Resources
2. Sizing
3. Architecture Overview
DOCUMENT LINKS

• ThingWorx Core 7.x Sizing Guide
  – https://support.ptc.com/view?im_dbkey=172249

• ThingWorx 7.x Core Deployment Sizing Calculator
  – https://support.ptc.com/view?im_dbkey=172248

• ThingWorx Architecture Overview
  – ...coming soon...
COMPLEMENTARY LIVEWORX PRESENTATIONS

• ThingWorx Platform Architecture
  – ThingWorx High Scale Ingest
    Wed. May 24 11:15am – 12:00pm

• Asset Connectivity
  – ThingWorx & Device Clouds
    Wed. May 24 11:15am – 12:00pm
  – ThingWorx and Device Management
    Wed. May 24 3:00pm – 3:45pm
  – ThingWorx and Connectivity
    Thur. May 25 11:15am – 12:00pm

• Security
  – IoT Security Challenges and How to Meet Them
    Wed. May 24 4:00pm – 4:45pm

• Project Management
  – Accelerating Your IoT Solution Journey from Exploring to Development to Deployment and Operation
    Wed. May 24 9:15am – 10:00am
SIZING
PURPOSE OF THE SIZING GUIDE AND CALCULATOR

- Standard method for platform sizing
- Quick and reasonable estimates during sales engagements
- Intended to support 70 - 80% of sizing questions
WHAT TO EXPECT FROM THE GUIDE

What it offers

• T-shirt Size estimate for ThingWorx and database
  – Small, medium, or large
  – On-premise and cloud sizing

• The need for connection servers

• Indicate if a DataStax system is warranted based on requirements

What it does not offer

• Scalability benchmarking
  – Imposed limits should not be construed as a platform ceiling

• Guidance on Platform scaling

• Guidance for DataStax sizing
# SIZING GUIDE - DATASET

<table>
<thead>
<tr>
<th></th>
<th>Extra small</th>
<th>Small</th>
<th>Medium</th>
<th>Large</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Things</td>
<td>10,000</td>
<td>30,000</td>
<td>100,000</td>
<td>250,000</td>
</tr>
<tr>
<td>Properties/thing</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Ingestion Rates</td>
<td></td>
<td></td>
<td>Increases during test run</td>
<td></td>
</tr>
<tr>
<td>Subscriptions</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Alerts</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>http requests</td>
<td></td>
<td></td>
<td>Increases during test run</td>
<td></td>
</tr>
</tbody>
</table>
SIZING PARAMETERS - USER

- **Number of Named Users**
  - Total number of authenticated users

- **User Concurrency**
  - Number of users making requests at the same time.
SIZING PARAMETERS – DATA INGESTION

• Number of Things  
  – Number of assets managed by ThingWorx  
  – Impacts platform memory requirements

• Number of Properties per Thing  
  – Volume of data sent to platform per submission

• Write Frequency  
  – How often data is sent  
  – Dependent on network connection  
  – Direct impact on database sizing
HOW TO USE THE GUIDE

1. Collect your ThingWorx usage requirements
   • Users and concurrency
   • Things, properties, and write frequency

2. Calculate key sizing criteria

<table>
<thead>
<tr>
<th>Key Sizing Parameter</th>
<th>Definition</th>
<th>Your Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
<td>Number of Connection servers to be added</td>
<td></td>
</tr>
<tr>
<td>HTTP Requests</td>
<td>Estimated concurrent user requests to ThingWorx</td>
<td></td>
</tr>
<tr>
<td>Thingcount</td>
<td>Estimated number of Things managed by ThingWorx</td>
<td></td>
</tr>
<tr>
<td>VS Queue</td>
<td>Estimated writes per second to ThingWorx</td>
<td></td>
</tr>
</tbody>
</table>

3. Compare sizing criteria to guidance
   • Determine whether Small, Medium, or Large

- Thingcount = 100,000 → Medium
- Connection Servers = 2 → Two connection servers
- VS queue rate = 6,600 wps → Medium (with PostgreSQL)
- HTTP requests = 100 → Small
HOW TO USE THE GUIDE

4. Match size to hardware estimates
   - ThingWorx

<table>
<thead>
<tr>
<th>Size</th>
<th>AWS EC2</th>
<th>Azure VM</th>
<th>On-premise CPU Cores</th>
<th>On-Premise RAM</th>
<th>Storage Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extra Small/H2</td>
<td>C4.xlarge</td>
<td>F4</td>
<td>4</td>
<td>7.5</td>
<td>750</td>
</tr>
<tr>
<td>Small/H2</td>
<td>C4.2xlarge</td>
<td>F8</td>
<td>8</td>
<td>15</td>
<td>1,000</td>
</tr>
<tr>
<td>Small</td>
<td>C4.2xlarge</td>
<td>F8</td>
<td>8</td>
<td>15</td>
<td>1,000</td>
</tr>
<tr>
<td>Medium</td>
<td>C4.4xlarge</td>
<td>F16</td>
<td>16</td>
<td>30</td>
<td>2,000</td>
</tr>
<tr>
<td>Large</td>
<td>C4.8xlarge</td>
<td></td>
<td>36</td>
<td>60</td>
<td>4,000</td>
</tr>
</tbody>
</table>

- PostgreSQL Database

<table>
<thead>
<tr>
<th>Size</th>
<th>AWS EC2</th>
<th>Azure VM</th>
<th>On-Premise CPU Cores</th>
<th>On-Premise RAM</th>
<th>SSD Storage (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small</td>
<td>C3.2xlarge</td>
<td>F8</td>
<td>8</td>
<td>15</td>
<td>160</td>
</tr>
<tr>
<td>Medium</td>
<td>C3.4xlarge</td>
<td>F16</td>
<td>16</td>
<td>30</td>
<td>320</td>
</tr>
<tr>
<td>Large</td>
<td>C3.8xlarge</td>
<td></td>
<td>32</td>
<td>60</td>
<td>640</td>
</tr>
</tbody>
</table>

5. Consider additional load impacts
   - High Availability requirements
   - File Vaulting
   - Subscriptions and events
ARCHITECTURE OVERVIEW
THINGWORX SOLUTION ARCHITECTURE

**Connect**
- IoT Clouds
- Gateway
- ThingWorx Industrial Connectivity
- Edge/OT Environments
- CAD & PLM, AEC & BIM, Digital Context

**Create & Manage**
- ThingWorx Composer
  - Model Definition
  - Analytics Definition
  - Business Logic

**Analyze & Engage**
- ThingWorx Mashup Builder
- REST APIs
- ThingWorx Analytics
- ThingWorx Studio

Persistence Providers
- H2 (Embedded)
- PostgreSQL
- MS SQL
- DSE
- SAP HANA

Secure Data Flow
- Database infrastructure (On premise or Cloud)
- JDBC Connector
- Integration Connectors

**ThingWorx Major Component**
- Sub-components
- Features & functionality
- Secured Architecture
- Secured Data flow

Desktop/Laptop
- Mobile
- AR/Wearables
- VR/Wearables
THINGWORX SOLUTION ARCHITECTURE

Connect
- IoT Clouds
- Gateway
- ThingWorx connection server
- Edge/OT Environments
- CAD & PLM, AEC & BIM Digital Context

Create & Manage
- ThingWorx Composer
  - ThingWorx Integration Builder
  - ThingWorx Runtime

Analyze & Engage
- ThingWorx Analytics
- ThingWorx Studio
- Integration Connectors
- Secure Data Flow
- ThingWorx Major Component
- Sub-components
- Features & Functionality
- Secured Architecture
- Secured Data flow

Persistence Providers
- H2 (Embedded)
- PostgreSQL
- MS SQL
- DSE
- SAP HANA

Security Components
- Business Logic
- Model Definition
- Analytics Definition
- Business/IT Systems

Remote Connectivity
- Desktop/Laptop
- Mobile
- AR/Wearables
- VR/Wearables
ARCHITECTURE OVERVIEW - BREAKDOWN

• **Things/Devices**
  – Connecting things, devices, assets, etc. to the ThingWorx platform.

• **Users/Clients**
  – Connecting users to the platform.

• **Platform**
  – Application tier. Content from devices is ingested, requests from users are answered, content is analyzed to generate alerts.

• **Database**
  – Contain the ThingWorx runtime model definitions and its persisted properties.
  – Contain tabular data. Rows of content within blogs, wikis, streams, ValueStreams, and Data Tables.
COMPONENTS – USERS / CLIENTS

- HTML/HTML5 Web Browser
  - Supports ThingWorx Composer
  - Supports ThingWorx Mashups

- Authentication Services configured to ThingWorx (planned in ThingWorx 8.0+)
ARCHITECTURE OVERVIEW - USERS / CLIENTS

Authentication Schemes

- Simple authentication direct with ThingWorx/Tomcat

- Industry standard SAML authentication process (planned for ThingWorx 8.0+)
COMPONENTS – PLATFORM

• ThingWorx Connection Server
  – Separate server application
  – Scalable connectivity over WebSockets
  – Redirects content to ThingWorx platform
  – Starting point: one per 50,000 connections
  – Needs sticky sessions when operating through load balancer

• ThingWorx Server
  – Has everything you need to make IoT work for your business
  – Java application running in an Apache Tomcat container
  – Device communication through WebSocket protocol
COMPONENTS – DATABASE

Supported Persistence Providers for Model and Stream content

- H2 (embedded)
- PostgreSQL
- SAP Hana (with ThingWorx 7.3+)
- Microsoft SQL Server (with ThingWorx 7.4+)

Supported Persistence Providers for Stream content only

- DataStax Enterprise
  - When more then 15,000 writes/second is needed
Monolithic or Basic system

- Typical system for
  - Development
  - functional QA
  - Training
  - Proof of concepts

- Imbedded H2 database is usually sufficient
REFERENCE ARCHITECTURE - PRODUCTION

Basic Production System

• Good for small and medium sized global system
• Good for medium and large manufacturing system
• Separated Database
• PTC recommends PostgreSQL, SAP Hana or MS SQL Server for production instances
REFERENCE ARCHITECTURE - ENTERPRISE

Enterprise System

- Designed to meet high data ingestion rates
- Addition of DataStax Enterprise for Stream content
- Addition of Connection Servers
COMPONENTS – HIGH AVAILABILITY

ThingWorx Components for HA

• Load Balancer
  – Routes traffic to the active ThingWorx server

• Apache ZooKeeper
  – Monitors ThingWorx availability and provides leader election
  – Minimum of three zNodes

Database Components for HA

• H2 – none for embedded version
• PostgreSQL – pgpool II
• SAP Hana – backups, storage and system replication
• SQL Server – Windows Server Failover Clustering
High Availability System (HA)

• Load balancers to direct to active platform

• Active/Passive approach for ThingWorx Platform

• ZooKeeper to elect active platform

• Apply HA requirements for chosen data providers
  – DSE is inherently HA by design
  – PostgreSQL using pgpool II and multiple nodes
Cloud Deployments

- Any of these deployments can work in a cloud environment.

- Take advantage of Cloud features
  - Elastic Load Balancing
  - RDS databases
Distributed / Federated System

- Aggregate data to another system
  - Distribute high workloads
  - Connect Corporate to independent regions

- Each ThingWorx Platform can operate in any previous architecture and still function as a federated system
WE WANT YOUR FEEDBACK

Please remember to complete your evaluation by selecting the session in your mobile app.
BACKUP SLIDES
COMPONENTS – THINGS / DEVICES

• ThingWorx Edge SDKs
  – Framework for communicating with ThingWorx from a remote location
  – Uses WebSockets protocol for communication
  – Allows persistent “AlwaysOn” connection (through firewalls)

• ThingWorx Edge Microserver
  – Edge SDK-based application installed on a remote device
  – Establishes AlwaysOn bidirectional communication with ThingWorx server
  – Enables secure communication between devices and ThingWorx server
ARCHITECTURE OVERVIEW – THINGS / DEVICES

1. IoT Cloud
   - AWS IoT
   - Azure IoT
   - Predix Connector
   - Aveda Connector
   - SAP HCP IoT

2. Embedded

3. Tethered

4. Networked

5. ThingWorx Industrial Connectivity

Legend:
- Optional Solution Component
- ThingWorx Connection Server
- Secured data flow
SIZING EXAMPLE – SMART CITY MONITORING

• **Scenario:** Monitoring the 100,000 water meters throughout a city. Each water meter reports 20 property values to ThingWorx every 5 minutes. There are up to 500 known users that will access ThingWorx, expecting up to 20% to be online simultaneously at times.

1. Collect requirements
   • Number of things: 100,000
   • Number of properties: 20
   • Write frequency: every 5 minutes
   • Estimated number of users: 500
   • Estimated max user concurrency: 20%
2. Calculate sizing criteria

- **Thingcount = 100,000**
- **Connection Servers = 100,000 / 50,000**
- **Connection Servers = 2**

- Transmission frequency = every 5 minutes = 288 writes/day = 0.0033 writes/sec
- Operations per second (OPS) = 100,000 * 0.0033 wps = 330 ops
- Property writes per second (PWS) = 330 ops * 20 properties = 6,600 wps
- **VS queue rate = 6,600 wps**

- User concurrency = 500 users * 0.20 concurrency rate = 100 concurrent users
- **HTTP requests = 100**
3. Compare sizing criteria to guidance

- Thingcount = 100,000 → Medium
- Connection Servers = 2 → Two connection servers
- VS queue rate = 6,600 wps → Medium (with PostgreSQL)
- HTTP requests = 100 → Small

Overall recommended sizes
Medium ThingWorx server
Medium PostgreSQL server
Two Connection servers
SIZING EXAMPLE – SMART CITY MONITORING

4. Match “medium” size to hardware estimates

- **ThingWorx Platform**

<table>
<thead>
<tr>
<th>Size</th>
<th>AWS EC2 Instance</th>
<th>Azure VM</th>
<th>On-premise CPU Cores</th>
<th>On-premise Memory (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Medium</td>
<td>C4.4xlarge</td>
<td>F16</td>
<td>16</td>
<td>30</td>
</tr>
</tbody>
</table>

- **PostgreSQL server**

<table>
<thead>
<tr>
<th>Size</th>
<th>AWS EC2 Instance</th>
<th>Azure VM</th>
<th>On-premise CPU Cores</th>
<th>On-premise Memory (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Medium</td>
<td>C3.4xlarge</td>
<td>F16</td>
<td>16</td>
<td>30</td>
</tr>
</tbody>
</table>

- **Two Connection Servers**

<table>
<thead>
<tr>
<th>Size</th>
<th>AWS EC2 Instance</th>
<th>Azure VM</th>
<th>On-premise CPU Cores</th>
<th>On-premise Memory (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Medium</td>
<td>C4.2xlarge</td>
<td>F8</td>
<td>8</td>
<td>15</td>
</tr>
</tbody>
</table>
5. Additional Load Impacts

- High Availability requirements?
- Other load from assets? (images, other files to be transferred, etc)
- What subscriptions and events are to be managed on the platform?