**Lab 201 – ScaleOut using MSMQ** (rev 1.3)

When the need arises to scale out a certain operation to multiple machines, after having already maxed out on the number of threads on the same machine, the Distributor feature of NServiceBus comes in handy. This feature is applicable only when using MSMQ as the transport of choice. The distributor can be thought of as a load balancer.

When scaling out, one shouldn’t have to resort to changing code. In NServiceBus, the same handler code can be made to scale out by using feature profiles: NServiceBus.Master or NServiceBus.Distributor and NServiceBus.Worker. For more information on profiles, see:

<http://particular.net/articles/more-on-profiles>

Clients are configured to send the messages to the Distributor/Master node endpoint. The distributor/master then load balances the messages it received to the worker nodes that are registered with it. As a standard NServiceBus process, the distributor maintains all the fault-tolerant and performance characteristics of NServiceBus but also is designed never to overwhelm anyof the worker nodes configured to receive work from it.

For more information see:

<http://particular.net/articles/load-balancing-with-the-distributor>

Lab Objectives

**In this lab you will learn**

* How to scale out the load on the endpoints when using MSMQ as the transport, without changing code, by simply changing configuration.
* How to set up an endpoint as either the master node or the distributor node, which will act as the load balancer.
* How to set up a node as a worker node, which will receive work from the master/distributor node.
* How the system load gets distributed by the master node to the registered worker nodes.

**NOTE**: The distributor is only applicable when using MSMQ as the message transport. It is not applicable when using other transports such as ActiveMQ, RabbitMQ, or SQLServer.

Lab Prerequisites

This lab already has these components pre-installed:

* Visual Studio 2012
* NServiceBus Infrastructure – DTC
* NServiceBus Infrastructure – MSMQ
* NServiceBus Infrastructure – Performance Counters

**NOTE**: To install NServiceBus infrastructure on your machines, use Powershell commandlets.

<http://particular.net/articles/managing-nservicebus-using-powershell>

Problem Definition

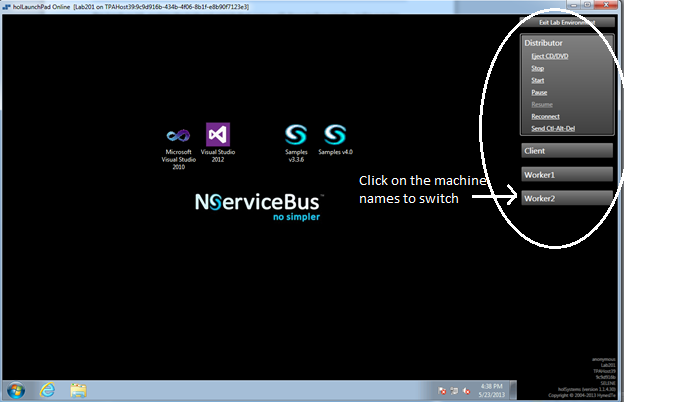
This hands on lab explains the ScaleOut sample that comes with NServiceBus samples. In this exercise, you will run the ScaleOut exercise and see how to scale this endpoint to multiple machines without changing code, by changing just the feature profile and minimal configuration.

Lab Machines

To see the ScaleOut in action, this lab is set up as four machines:

|  |  |
| --- | --- |
| Machine Name | Purpose |
| Distributor | This VM hosts the Retail.Orders.Handler endpoint, which runs either in the master profile or the distributor profile, which acts as the load balancer. |
| Client | This VM hosts the Retail.Orders.Client endpoint, which simulates the load to the distributor. |
| Worker1 | This VM hosts the same Retail.Orders.Handler endpoint running in the distributor machine, and instead runs it in the worker profile. This endpoint handles the load from the distributor. |
| Worker2 | This VM hosts the same Retail.Orders.Handler endpoint running in the distributor machine, and instead runs it in the worker profile. This endpoint handles the load from the distributor. |

To switch machines, click the buttons labeled “Client”, “Worker1”, or “Worker2” that are listed on the right top corner of the lab.



**Firewall settings:**

The following programs have been set as Allowed in the firewall configuration:

* All machines are allowed for File& Print Sharing, ICMP, Core Networking, MSMQ, Network Discovery, DTC and Visual Studio.
* All machines have DTC enabled and allow ports 5000-6000 and other settings configured as shown here:

<http://particular.net/articles/transactions-message-processing>

* The distributor machine has RavenDB port 8080 enabled.
* All of the VMs in this lab have unique MSMQ QMId in the registry and also a unique CID for the MSDTC in the registry, as described in the following articles:

<http://blogs.msdn.com/b/johnbreakwell/archive/2007/02/06/msmq-prefers-to-be-unique.aspx>

<http://www.wadewegner.com/2007/08/warning-the-cid-values-for-both-test-machines-are-the-same/>

**Estimated time to complete:**

60 mins.

Terms of Use

By using this Hands-on Lab, you agree to the following terms:

The technology/functionality described in this Hands-on Lab is provided by NServiceBus in a “sandbox” testing environment for purposes of obtaining your feedback and to provide you with a learning experience. You may only use the Hands-on Lab to evaluate such technology features and functionality and provide feedback to NServiceBus. You may not use it for any other purpose. You may not modify, copy, distribute, transmit, display, perform, reproduce, publish, license, create derivative works from, transfer, or sell this Hands-on Lab or any portion thereof.

COPYING OR REPRODUCTION OF THE HANDS-ON LAB (OR ANY PORTION OF IT) TO ANY OTHER SERVER OR LOCATION FOR FURTHER REPRODUCTION OR REDISTRIBUTION IS EXPRESSLY PROHIBITED.

THIS HANDS-ONLAB PROVIDES CERTAIN SOFTWARE TECHNOLOGY/PRODUCT FEATURES AND FUNCTIONALITY, INCLUDING POTENTIAL NEW FEATURES AND CONCEPTS, IN A SIMULATED ENVIRONMENT WITHOUT COMPLEX SET-UP OR INSTALLATION FOR THE PURPOSE DESCRIBED ABOVE. THE TECHNOLOGY/CONCEPTS REPRESENTED IN THIS HANDS-ON LAB MAY NOT REPRESENT FULL FEATURE FUNCTIONALITY AND MAY NOT WORK THE WAY A FINAL VERSION MAY WORK. WE ALSO MAY NOT RELEASE A FINAL VERSION OF SUCH FEATURES OR CONCEPTS. YOUR EXPERIENCE WITH USING SUCH FEATURES AND FUNCITONALITY IN A PHYSICAL ENVIRONMENT MAY ALSO BE DIFFERENT.

**FEEDBACK**. If you give feedback about the technology features, functionality and/or concepts described in this Hands-on Lab to NServiceBus, you give to NServiceBus, without charge, the right to use, share and commercialize your feedback in any way and for any purpose. You also give to third parties, without charge, any patent rights needed for their products, technologies and services to use or interface with any specific parts of a NServiceBus software or service that includes the feedback. You will not give feedback that is subject to a license that requires NServiceBus to license its software or documentation to third parties because we include your feedback in them. These rights survive this agreement.

NServiceBus Ltd. HEREBY DISCLAIMS ALL WARRANTIES AND CONDITIONS WITH REGARD TO THE HANDS-ON LAB , INCLUDING ALL WARRANTIES AND CONDITIONS OF MERCHANTABILITY, WHETHER EXPRESS, IMPLIED OR STATUTORY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. NServiceBus DOES NOT MAKE ANY ASSURANCES OR REPRESENTATIONS WITH REGARD TO THE ACCURACY OF THE RESULTS, OUTPUT THAT DERIVES FROM USE OF THE VIRTUAL LAB, OR SUITABILITY OF THE INFORMATION CONTAINED IN THE VIRTUAL LAB FOR ANY PURPOSE.

Contents

[Exercise 1: Scale out using MSMQ with master/worker 6](#_Toc405807379)

[Task 4: Start the Retail.Orders.Handler endpoint to startup as master 6](#_Toc405807380)

[Task 5: Deploy Retail.Orders.Sender to the client machine 7](#_Toc405807381)

[Task 6: Deploy Retail.Orders.Handler to the Worker1 machine 8](#_Toc405807382)

[Task 7: Deploy Retail.Orders.Handler to the Worker2 machine 9](#_Toc405807383)

[Task 8: Simulate load and watch the processing 9](#_Toc405807384)

[Exercise 2: Switch to distributor/worker profiles 9](#_Toc405807385)

[Task 1: Run the Retail.Orders.Handler using the distributor profile 9](#_Toc405807386)

[Task 2: Simulate load and watch the processing 10](#_Toc405807387)

[Completed Lab Solution 10](#_Toc405807388)

# Exercise 1: Scale out using MSMQ with master/worker

Switch to the Distributor Machine (by clicking on Distributor on the right side of the lab window, where all the servers are listed) if the Distributor machine is not the currently active machine.

Download the starter solution, **Lab201.zip** from (<https://github.com/Particular/HandsOnLabs/releases>) and extract it to **C:\Hands on Labs\Exercises**.

Right click on the solution and select, **Enable Package Restore** and click **Yes** to the prompt to restore the nuget packages.

Open a new instance of Visual Studio 2012 and make sure to run it as an administrator and open **Retail.Scaleout** solution from **C:\Hands on Labs\Exercises\Lab201\Exercise** .

In this solution the Retail.Orders.Sender sends 10 PlaceOrder commands whenever the Enter key is pressed. The Retail.Orders.Handler processes the command and publishes an event and the Sender is subscribed to the OrderPlaced event. Our goal is to ScaleOut the Retail.Orders.Handler endpoint so that it can handle the load.

To test the solution, start the Retail.Orders.Handler endpoint. To do this, right click on Retail.Orders.Handler -> Debug -> Start New Instance

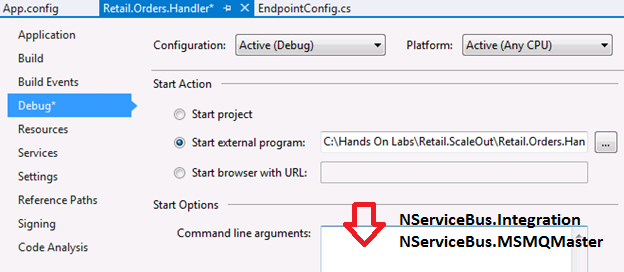
Start the Retail.Orders.Sender endpoint. To do this, right click on Retail.Orders.Handler -> Debug -> Start New Instance and click Enter and this should send the PlaceOrder command to the handler and receive the OrderPlaced event.

## Task 4: Start the Retail.Orders.Handler endpoint to startup as master

1. Stop the endpoints.
2. Install the distributor nuget package. In Package Manager Console (Tools -> Nuget Package Manager -> Package Manager Console) type:

**Install-package NServiceBus.Distributor.Msmq Retail.Orders.Handler**

1. Set up the command line argument to include NServiceBus.Integration and NServiceBus.MSMQMaster as shown in the Project properties -> Debug for Retail.Orders.Handler:



***About the NServiceBus.MSMQMaster profile:*** When running under the master profile, when the endpoint starts up as a load balancer, this node also processes the messages that are received, in addition to delegating work to the other worker nodes.

***About the NServiceBus.Integration Profile*:** When running the endpoint from the command line, using the NServiceBus.Integration profile allows the queues to be created on startup

1. Modify app.config and ensure that the configuration is set up appropriately to scale up by setting the MaximumConcurrency level defined in TransportConfig section.

Add TransportConfig as shown below in your app.config.

Adjust the MaximumConcurrencyLevel in the TransportConfig section, as shown to process five messages at a time:

<?xml version="1.0" encoding="utf-8" ?>

<configuration>

<configSections>

<section name="MessageForwardingInCaseOfFaultConfig" type="NServiceBus.Config.MessageForwardingInCaseOfFaultConfig, NServiceBus.Core" />

<section name="TransportConfig" type="NServiceBus.Config.TransportConfig, NServiceBus.Core"/>

</configSections>

<MessageForwardingInCaseOfFaultConfig ErrorQueue="error"/>

<TransportConfig MaxRetries="2" MaximumConcurrencyLevel="5" MaximumMessageThroughputPerSecond="0"/>

</configuration>

1. Start the Retail.Orders.Handler endpoint. This should now run using the Master profile.
2. The binaries will be available to the other virtual machines via the File Share [\\Distributor\Exercises](file:///\\Distributor\Exercises), in order to facilitate deployment of the code to the other machines.

## Task 5: Deploy Retail.Orders.Sender to the client machine

1. Switch to the client machine by clicking Client on the right side of the lab window, where the servers are listed.
2. On this machine, copy the binaries from

[\\Distributor\Exercises\Lab201\Exercise\Retail.Orders.Sender\bin\debug](file:///\\Distributor\Exercises\Lab201\Exercise\Retail.Orders.Sender\bin\debug) to C:\Hands On Labs\Deployment.

1. Change the Retail.Orders.Sender.dll.config in C:\Hands On Labs\Deployment, so that the messages are sent to the master node, i.e. the Distributor:

<UnicastBusConfig>

<MessageEndpointMappings>

<!--To register a specific type in an assembly -->

<add Assembly="Retail.Orders.Messages" Type="Retail.Orders.Messages.Events.OrderPlaced" Endpoint="Retail.Orders.Handler@Distributor" />

<add Assembly="Retail.Orders.Messages" Type="Retail.Orders.Messages.Commands.PlaceOrder" Endpoint="Retail.Orders.Handler@Distributor" />

</MessageEndpointMappings>

</UnicastBusConfig>

1. To start the Sender endpoint, Open an administrative command prompt, change the directory to the correct location (i.e., C:\Hands on Labs\Deployment). Start the sender in the Integration profile as follows:

**NServiceBus.Host NServiceBus.Integration**

## Task 6: Deploy Retail.Orders.Handler to the Worker1 machine

1. Switch to the **Worker1** machine by clicking Worker1 on the right side of the lab window, where the servers are listed.
2. On this machine, copy the binaries from [\\Distributor\Exercises\Lab201\Exercise\Retail.Orders.Handler\bin\debug](file:///\\Distributor\Exercises\Lab201\Exercise\Retail.Orders.Handler\bin\debug) to C:\Hands on Labs\Deployment.\\
3. Modify the Retail.Orders.Handler.dll.config in this location and ensure that the configuration is set up appropriately in app.config as shown to:

* Ensure that the master node configuration is properly specified, as shown below.
* Specify the distributor control address and the application queue located at the master node in UnicastBusConfig.

<?xml version="1.0" encoding="utf-8" ?>

<configuration>

<configSections>

<section name="MessageForwardingInCaseOfFaultConfig" type="NServiceBus.Config.MessageForwardingInCaseOfFaultConfig, NServiceBus.Core" />

<section name="TransportConfig" type="NServiceBus.Config.TransportConfig, NServiceBus.Core"/>

<section name="UnicastBusConfig" type="NServiceBus.Config.UnicastBusConfig, NServiceBus.Core" />

<section name="MasterNodeConfig" type="NServiceBus.Config.MasterNodeConfig, NServiceBus.Core" />

</configSections>

<MasterNodeConfig Node="Distributor"/>

<MessageForwardingInCaseOfFaultConfig ErrorQueue="error"/>

<TransportConfig MaximumConcurrencyLevel="5" MaxRetries="2"/>

<UnicastBusConfig DistributorControlAddress="retail.orders.handler.distributor.control@Distributor"

DistributorDataAddress="retail.orders.handler@Distributor">

</UnicastBusConfig>

</configuration>

1. To start the endpoint, Open an administrative command prompt, change the directory to C:\Hands on Labs\Deployment Start the endpoint in the worker mode by typing

**NServiceBus.Host NServiceBus.Integration NServiceBus.MSMQWorker**

## Task 7: Deploy Retail.Orders.Handler to the Worker2 machine

1. Switch to the Worker2 machine by clicking Worker2 on the right side of the lab window, where the servers are listed.
2. Copy the binaries from [\\Worker1\Deployment](file:///\\Worker1\Deployment) to C:\Hands on Labs\Deployment
3. To start the endpoint, open an administrative command prompt, change the directory to C:\Hands on Labs\Deployment and start the endpoint in the worker mode by typing:

**NServiceBus.Host NServiceBus.Integration NServiceBus.MSMQWorker**

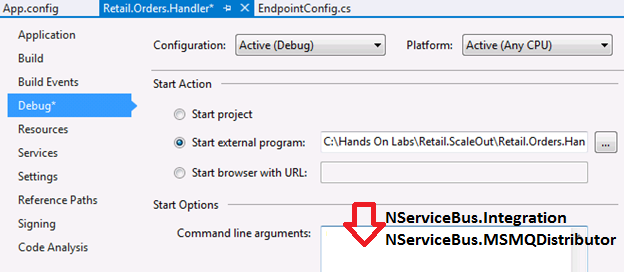
## Task 8: Simulate load and watch the processing

1. Switch to the client machine by clicking Client on the right side, where the servers are listed.
2. Press the Enter key a few times to simulate a load of messages that are sent from the Retail.Orders.Sender running on the client machine.
3. Switch to the Distributor machine where the Retail.Orders.Handler host is running using the NServiceBus.Master profile and watch the endpoint process messages in addition to distributing the load.
4. Switch to the Worker1 machine and watch the Retail.Orders.Handler.
5. Switch to the Worker2 machine and watch the Retail.Orders.Handler.
6. Switch to the client machine and wait to receive the responses for all messages sent.

# Exercise 2: Switch to distributor/worker profiles

## Task 1: Run the Retail.Orders.Handler using the distributor profile

1. Switch to the distributor machine by clicking Distributor on the right side where the servers are listed.
2. To stop the endpoint, press Ctrl + C, or stop debugging on the VS2012 IDE.
3. Add the distributor profile instead of the master profile as the command line arguments in Project Properties for Retail.Orders.Handler, as shown:



***About the NServiceBus.Distributor profile*:**  When running in this profile, NServiceBus starts the endpoint in the distributor mode, but does not enlist itself as a worker.

## Task 2: Simulate load and watch the processing

1. Switch to the client machine by clicking Client on the right side where the servers are listed.
2. Press the Enter key a few times to simulate a load.
3. Switch to the distributor machine and watch the Retail.Orders.Handler. Note that it does not handle any messages; instead, it acts as a pure load balancer.
4. Switch to the Worker1 machine and watch the Retail.Orders.Handler.
5. Switch to the Worker2 machine and watch the Retail.Orders.Handler.
6. Switch to the client machine and wait to receive the responses for the PlaceOrder command messages that were sent.

# Completed Lab Solution

The complete solution using Version 5.x and this manual is here:

<https://github.com/Particular/HandsOnLabs/tree/master/Lab201>

The complete solution for this exercise that uses NServiceBus version ***4.x*** can be found under C:\Hands on Labs\Completed Solution