



(https://www.codecentric.de/)

DE (HTTPS://BLOG.CODECENTRIC.DE/EN/CATEGORY/AGILE-EN/) SERVICES (HTTPS://WWW.CODECENTRIC.DE/EN/DOCKER/)

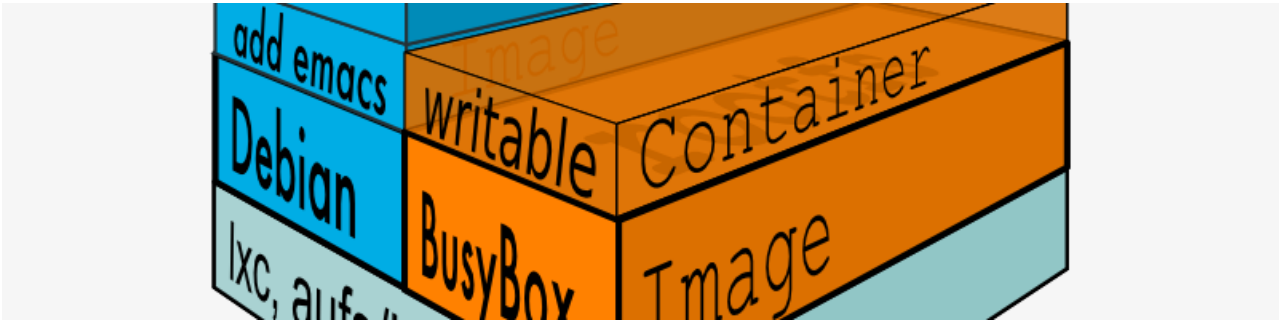
# codecentric Blog (https://blog.codecentric.de/)

POSTS

- NEW (/EN/)
- AGILE (HTTPS://BLOG.CODECENTRIC.DE/EN/CATEGORY/AGILE-EN/)
- ARCHITECTURE (HTTPS://BLOG.CODECENTRIC.DE/EN/CATEGORY/ARCHITECTURE/)
- JAVA (HTTPS://BLOG.CODECENTRIC.DE/EN/CATEGORY/JAVA-EN/)
- PERFORMANCE (HTTPS://BLOG.CODECENTRIC.DE/EN/CATEGORY/PERFORMANCE-EN/)
- CONTINUOUS DELIVERY (HTTPS://BLOG.CODECENTRIC.DE/EN/CATEGORY/CONTINUOUS-DELIVERY-AGILE-EN/)
- MICROSERVICES (HTTPS://BLOG.CODECENTRIC.DE/EN/CATEGORY/ARCHITECTURE/MICROSERVICES-ARCHITECTURE/)
- CLOUD (HTTPS://BLOG.CODECENTRIC.DE/EN/CATEGORY/ARCHITECTURE/CLOUD-ARCHITECTURE/)

RSS-Feed (

Overview (https://blog.codecentric.de/en/category/infrastructure-en/)



(<https://blog.codecentric.de/en/2014/10/dockerizing-mule-esb-enterprise/>)

# Dockerizing Mule ESB Enterprise

## (<https://blog.codecentric.de/en/2014/10/dockerizing-mule-esb-enterprise/>)

10/23/14 by **Conrad Pöpke** (<https://blog.codecentric.de/en/author/conrad-poepke/>)

2 Comments (<https://blog.codecentric.de/en/2014/10/dockerizing-mule-esb-enterprise/#comments>)

### Abstract

In the last years the industry has identified the demand in development and operation automation which led to a growing number of tools in this area. We have seen the industrialization of virtualization which made DevOps automation possible. The rise of provisioning tools such as Chef, Puppet or the newcomers Ansible and YADT. Hypervisor solutions to optimize and isolate bundled systems such as Docker or SmartOS or enterprise wide monitoring solutions such as AppDynamics or Nagios. All these different parts are subsumed up under the buzz word DevOps.

Now in the context of a project which required a development environment and a test environment for end-to-end system tests we had the need to create multiple MuleEE instances at once. The DevOps approach and tools were a natural choice. We decided to use Docker to resourcefully create multiple instances to keep the overhead in system tests as low as possible. During our research we only found Mule Community Edition Docker images for the creation of virtualized Mule instances. Hence my motivation arose to provide a enterprise version.

In this blog post we will depict how you can dockerize your own Mule ESB Enterprise for an enterprise environment, under the assumption that a proper license and MuleEE version is provided.

### Related Work

There exist already two Docker images in the Docker Hub which provide a bundled image

containing Mule ESB. Unfortunately they only contain the Community Edition:

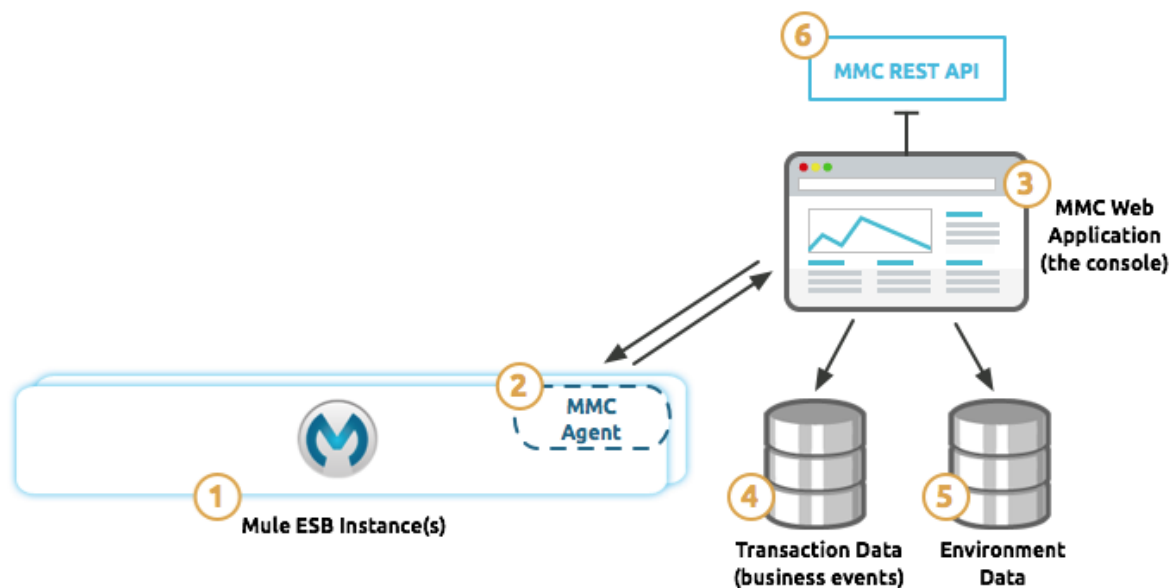
- [codingtony/mule](https://registry.hub.docker.com/u/codingtony/mule/) (<https://registry.hub.docker.com/u/codingtony/mule/>)
- [pr3d4t0r/mule](https://registry.hub.docker.com/u/pr3d4t0r/mule/) (<https://registry.hub.docker.com/u/pr3d4t0r/mule/>)

## Mule ESB infrastructure

In a Mule ESB enterprise deployment scenario we can have two types of nodes:

- Standalone MuleEE nodes which provide the Mule ESB functionality and can be clustered for high availability
- And a Mule Management Console node which provides an administration GUI for deploying, monitoring and managing the standalone MuleEE servers and it's applications

As depicted in the following graphic, the MMC can be deployed separately from the standalone MuleEE servers. The deployed MMC Agent on the MuleEE servers will help the MMC to auto discover the available instances.






(<https://blog.codecentric.de/files/2014/10/MMC-architecture.png>)

Caption 1: Architecture of the Mule Management Console (Source:<http://www.mulesoft.org/documentation/display/current>)

*/Architecture+of+the+Mule+Management+Console)*

Extract from the documentation:

 (https://blog.codecentric.de/files/2014/10/MMC-1.png)	The Mule ESB instances monitored by MMC. These can be standalone or embedded instances, or clusters.
	<p>The MMC agent contained in the Mule instance, which is responsible for:</p> <ul style="list-style-type: none"><li>• facilitating data transfer between the Mule instance and the MMC</li><li>• applying changes (i.e. thread pools, and configuration file changes) to the Mule applications</li></ul>
 (https://blog.codecentric.de/files/2014/10/MMC-3.png)	<p>The MMC instance, the Web-based interface that interacts with Mule through:</p> <ul style="list-style-type: none"><li>• the MMC agent</li><li>• JMX functionality exposed by Mule. (For more information on JMX management, see the JMX Management (<a href="http://www.mulesoft.org/documentation/display/current/JMX+Management">http://www.mulesoft.org/documentation/display/current/JMX+Management</a>) page)</li></ul> <p>The MMC instance is a Web application packaged as a WAR file that executes from within your Web servlet container (i.e. Tomcat, JBoss, etc.).</p>

## Build a Standalone Mule ESB Docker image

There are various good articles on the internet introducing Docker. Therefore we continue under the assumption that the reader already has some experience with Docker.

### Building the Docker base image

We created a Dockerfile which scripts the build steps for a Docker base image. It first takes the trial distribution package containing MuleEE and MMC, extracts the files, installs a Mule license – which needs to be provided by the user –, removes unnecessary content and configures the ports of the Docker base image. This Dockerfile can be obtained from GitHub (<https://github.com/cpoepke/docker/blob/master/mule-ee>) and a checkout will be assumed as a precondition.

Due to restrictions of the Enterprise version, the Docker image needs to be set up for individual

usage beforehand. It is required to:

- provide the Enterprise standalone server from Mulesoft. In this Dockerfile we assume the downloaded trial version from mulesoft.com (<http://www.mulesoft.com/mule-esb-enterprise-30-day-trial>), located in the same folder as the Dockerfile.
- provide the Enterprise license file from Mulesoft, located in the same folder as the Dockerfile.

Hence the prepared directory should look like this:

- ./Dockerfile
- ./mmc-distribution-mule-console-bundle-3.5.1.zip
- ./mule-ee-license.lic

(<https://github.com/cpoepke/docker#building-and-tag-the-docker-base-image>)Now we can build and tag the Docker base image:

```
docker build --tag="mule-ee" .
```

(<https://github.com/cpoepke/docker#container-types>)Image types

There are two ways now to use the Docker base image, depending on the overall scenario:

- you can use the base images to startup and create – in a classical operations sense – multiple standalone Mule ESB instances and one MMC instance. These can be used as a cluster as usual with hot deployment over MMC etc.
- or – which we recommend – create an Docker image for each Mule ESB application to isolate applications from each other and this way startup and create multiple standalone Mule ESB instances and one MMC instance. This might be a an option in a Micro Services or SOA scenario where you want more explicit control over your container. Please keep in this scenario your license in mind.

(<https://github.com/cpoepke/docker#app-specific-container-image>)App specific container image

Based on the MuleEE Docker base image, an application specific image can be build. The corresponding Dockerfile can look like this, the concrete realization is passed on to the reader:

```
FROM          mule-ee:latest
.
.
ADD           mule-app/target/mule-app-1.0.0-SNAPSHOT.zip /opt/mule-standa
```

Build application specific Docker image:

```
docker build --tag="my-mule-app-image" .
```

## Build a Mule Management Console Docker image

After building the base Docker image, we require a MMC specific Docker image.

As for the previous Docker image a Dockerfile was prepared which can be obtained from GitHub (<https://github.com/cpoepke/docker/blob/master/mule-mmc>) and again a checkout will be assumed as a precondition. Furthermore it is required to provide the Mule Management Console from Mulesoft. We assume the downloaded trial version from [mulesoft.com](http://www.mulesoft.com) (<http://www.mulesoft.com/mule-esb-enterprise-30-day-trial>), located in the same folder as the Dockerfile.

Hence the prepared directory should look like this:

- ./Dockerfile
- ./mmc-distribution-mule-console-bundle-3.5.1.zip

(<https://github.com/cpoepke/docker#building-and-tag-the-docker-base-image>) Afterwards we can build and tag the Docker image:

```
docker build --tag="mule-mmc" .
```

## Testdriving the dockerized Mule ESB infrastructure

Now after the creation of both image types or a couple of application specific images we will take them out for a spin.

For example we start two standalone Mule ESB Enterprise instances:

```
docker run -t -i --name='mule-ee-node1' mule-ee
docker run -t -i --name='mule-ee-node2' mule-ee
```

To start an app specific image use the following command:

```
docker run -t -i --name='mule-app-nodeX' my-mule-app-image
```

And we start a Mule MMC instance:

```
docker run -t -i -p 8585:8585 --name='mule-mmc-node' mule-mmc
```

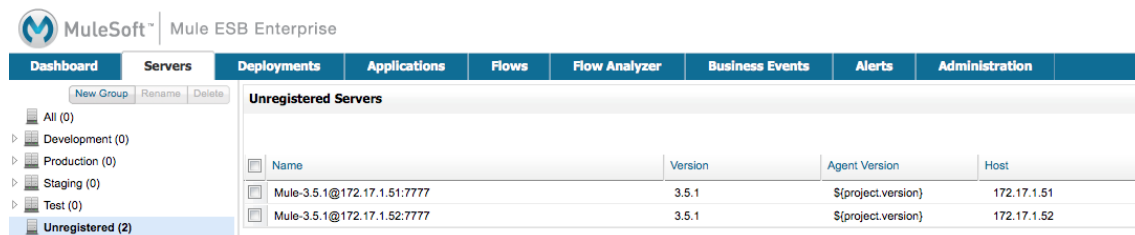
“ Notice: On OS X the VBox from boot2docker requires port forwarding from the host -> VBox -> Docker

```
boot2docker ssh -L 8585:localhost:8585
```

Now we can take a look on the running Docker instances and see two MuleEE nodes and the MMC node running:

```
cpoepke:mule-mmc cpoepke$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
f77b8b2bcabd        mule-ee:latest     /bin/sh -c /opt/mul     2 days ago         Up
500499fa5054        mule-ee:latest     /bin/sh -c /opt/mul     2 days ago         Up
a23cfc4c9df0        mule-mmc:latest    /bin/sh -c /opt/mmc     2 days ago         Up
```

And as expected entering <http://localhost:8585/mmc-3.5.1> we will land us on the MMC login screen. Furthermore we can now discover that our MuleEE nodes have been auto discovered by the MMC node. Hence this verifies that the connection between the nodes themselves is established:



(<https://blog.codecentric.de/files/2014/10/MMC-Mule-ESB-instances.png>)

Caption 2: The Mule Management Console showing the auto discovered MuleEE instances

## Conclusion

In this blog post we have shown how a MuleEE Docker base images and a MMC Docker images can be created. These can be used to ease the creation of developer, test or even production environments depending on the development and operations strategy of your company.

I am looking forward to your comments and lively discussions!

Log Management for Spring Boot Applications with Logstash, Elasticsearch and Kibana

Conrad Pöpke (<https://blog.codecentric.de/en/author/conrad-poepke/>)



(http://reddit.com  
/submit?url=https:  
//blog.codecentric.de  
(http://www.kodl.de/2014/10/dockerizing-  
(http://www.stumbleupon.com  
(http://www.budgetable.com/19/http://www.subling.comhttps:  
/sharer.php?u=https://blog.codecentric.de/submit?url=https:  
//blog.codecentric.de/2014/10/dockerizing-mule-enterprise-esb/  
/en/2014/en/2014/en/2014/10/dockerizing-mule-enterprise-esb-  
/10/dockerizing-mule-enterprise-esb-  
mule-esb-enterprise-esb-enterprise-esb-enterprise-esb-enterpris  
enterprise-enterprise-enterprise-enterprise-enterprise-enterpris  
se)enterpris(8.0text=DockerizingMuleESB=Enterprise%20Mule%20ESB%20Enterpris

Post by **Conrad Pöpke**



## ESB

Mule ESB Testing (Part 3/3): System End-to-End Testing with Docker (<https://blog.codecentric.de/en/2015/06/mule-esb-testing-part-33-system-end-to-end-testing-with-docker/>)

## More content about **Infrastructure**

### INFRASTRUCTURE

Highly available VIPs on OpenStack VMs with VRRP (<https://blog.codecentric.de/en/2016/11/highly-available-vips-openstack-vms-vrrp/>)

### INFRASTRUCTURE

Automatic Testing of Logstash Configuration (<https://blog.codecentric.de/en/2016/06/automatic-testing-logstash-configuration/>)

## Kommentare



29. October 2014 (<https://blog.codecentric.de/en/2014/10/dockerizing-mule-esb-enterprise/#comment-249475>) von **Yamen**

(<http://www.sixtree.com.au>)

Very nice Conrad! Something we are just now working on, so great timing.

Do you have any feel yet for the overhead (probably memory only) of running 'one Docker per app' as opposed to 'one Docker per container'? And in the 'Docker per container' scenario, have you decided whether to mount apps/conf/log directories to the host, or include in the container?

These are the kinds of decisions we're currently wrestling with, and it would be good to get some insight from someone who is there already.

Reply (<https://blog.codecentric.de/en/2014/10/dockerizing-mule-esb-enterprise/?replytocom=249475#respond>)



30. October 2014 (<https://blog.codecentric.de/en/2014/10/dockerizing-mule-esb-enterprise/#comment-250137>) von

**Conrad Pöpke** (<https://cpoepke.wordpress.com>)

Good d'mate!

Thank you for your comment.

I think you should ignore the overhead and put the emphasis more on your Continuous Integration/Delivery strategy and infrastructure landscape, because buying hardware resources is today not a issue. Depending on your strategy you can choose if you want to deploy your apps via the MMC (btw. there is a great Maven plugin (<http://search.maven.org/#artifactdetails%7Ccom.github.nicholasastuart%7Cmule-mmc-rest-plugin%7C1.2.0%7Cmaven-plugin>) for that) or by handling Docker containers.

Regarding the mounting of log file directories I would prefer to use the log directory in the container which is isolated from other containers this way and use something like logstash or graylog2 to send your log messages to a central logging collector such as Elasticsearch with Kibana or something similar. Just take a look here (<https://blog.codecentric.de/en/2014/10/log-management-spring-boot-applications-logstash-elasticsearch-kibana/>).

Cheers,

Conrad

Reply (<https://blog.codecentric.de/en/2014/10/dockerizing-mule-esb-enterprise/?replytocom=250137#respond>)

## Comment

Nachricht

Name

E-Mail

SUBMIT

**Notify me of followup comments via e-mail**

## Newsletter

E-Mail Adresse

SUBMIT

IMPRINT ([HTTPS://WWW.CODECENTRIC.DE/IMPRESSUM-DATENSCHUTZ/](https://www.codecentric.de/impressum-datenschutz/))

PRIVACY POLICY ([HTTPS://WWW.CODECENTRIC.DE/TERMS-AND-CONDITIONS/](https://www.codecentric.de/terms-and-conditions/))

CONTACT ([HTTPS://WWW.CODECENTRIC.DE/UEBER-CODECENTRIC/KONTAKT/](https://www.codecentric.de/ueber-codecentric/kontakt/))



(<https://www.facebook.com/codecentric>)



(<https://twitter.com/codecentric>)