OrderFlow System Deployment

OrderFlow Ltd.

Document Version: 4.2.4

Document Built: 2024-02-16

Introduction

OrderFlow is an enterprise-strength order processing and warehouse management system (WMS) with a wealth of configuration options and implementation possibilities.

This document provides details of how to deploy and configure an OrderFlow system, including what information to gather for the commissioning process.

Audience

The intended audience of this document is:

- business leaders who are considering OrderFlow as a solution to the needs of their organisation
- potential users and administrators of the OrderFlow system
- designers and developers of the OrderFlow system
- those responsible for supporting and deploying OrderFlow

OrderFlow Environment

OrderFlow is a modular, layered, multi-interface application, which exposes desktop, mobile and handheld user interfaces (via HTTP or HTTPS), accessible from a web browser. It also exposes an XML over HTTP application programming interface (API).

It is typically deployed within its own standalone Jetty web server, but can also be deployed to run within a separate web server (e.g. Apache Tomcat).
It is almost exclusively deployed in Linux-based environments, most commonly using Debian or Ubuntu distributions.

This section details the environment in which OrderFlow runs in greater detail, and provides details on how to set up the environment for OrderFlow deployment.

For more details of the technical architecture of the OrderFlow system, see the OrderFlow Technical Architecture document.

Hardware Resources

OrderFlow is typically deployed in a 'cloud' environment, i.e. on a virtual server within an internet-based hosting service. Such environments are usually dynamic in terms of the resources that can be made available to the virtual server, so it makes little sense to dictate the hardware that is required for OrderFlow to run.

Instead, other resources such as system memory, CPU speed and disk space *can* be specified. These resources will be required in different amounts for different customers, since the system usage will vary depending upon order volumes, user numbers, warehouse processes etc. For example, a paper-based warehouse may require fewer resources than a warehouse that uses handheld devices. Disk space requirements will be influenced by the database location(s) and any customer requirements on log file rotation.

The following ranges are typical for live deployments of OrderFlow:

- **System Memory** - from 2Gb to 65Gb. This is the memory available to the virtual machine that hosts OrderFlow.
- **JVM Heap memory** - from 800Mb to 7.5Gb. This is the memory available to the Java Virtual Machine, which is exclusively used by OrderFlow.
- **CPU** - anything from dual-core 2.67Ghz to 24-core 1.9GHz processors. The amount of CPU power a virtual machine has a directly-proportional effect on the responsiveness of the OrderFlow system. More users will require more CPU power.

Operating System

Although OrderFlow runs within a platform-agnostic Java Virtual Machine (within a *Jetty* web server), most OrderFlow deployments are on Linux-based operating systems. This is for the following reasons:

- Simplicity
- Consistency (with both the development environment and in between deployed environments)
- Cost

For deployments that we host (via a hosting provider), we use the latest stable ubuntu release. In customer-hosted environments, the choice of operating system lies with the customer, although we do recommend ubuntu to standardise our support.

To deploy OrderFlow, there needs to be an **rtdadmin** user created. This can be done (by the `root` user) by issuing the following command:

```
adduser rtdadmin
```

Once created, the password for this user should be changed (using the following command when logged in as `rtdadmin`), to a value that is recorded in RemoteMan.

```
sudo passwd
```

> **Note**
>
> **RemoteMan** is OrderFlow's configuration and monitoring system. Further details can be found in the RemoteMan Guide.

Once created the "rtdadmin" user should either be given 'sudo' access or if this is not appropriate be configured with the ability to:

- read,write,delete,execute permissions to the OrderFlow home directory (typically /usr/share/orderflow)
- The ability to start and stop the OrderFlow system through the init.d scripts

Software

OrderFlow is almost exclusively deployed within its own standalone Jetty web server. Therefore OrderFlow's system requirements are as follows:

Java Runtime Environment

The **Java Runtime Environment** currently needs to be compatible with version **8**, either OpenJDK or Oracle. This applies for OrderFlow versions to 4.2.1. From OrderFlow 4.2.2 onwards, *Java 11* is recommended for new environments but Java 8 is also supported.

To install version 8 of the **OpenJDK** Java Runtime Environment, run the following commands:

```
sudo apt-get update
sudo apt-get install openjdk-8-jre
```

To install OpenJDK 11, instead run:

```
sudo apt-get update
sudo apt-get install openjdk-11-jre
```

To install version 8 of the **Oracle** Java Runtime Environment, download the JRE (or Server JRE) from the Java SE 8 Archive Downloads page. Follow the instructions from either Oracle or WikiHow to install.

The JRE version in use can (currently) easily be switched by using the following command:

```
sudo update-alternatives --config java
```

From OrderFlow 4.2.2 onwards, Java 11 is recommeded

STRONG ENCRYPTION

In order to support strong encryption in some Linux environments, the Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy files need to be put in `lib/security/` under the JRE home directory.

We hold local copies of these files (local_policy.jar and US_export_policy.jar) in our source control system.

MariaDB

MariaDB is a drop-in replacement for MySQL. It is easy to install, offers many speed and performance improvements, and is easy to integrate into most MySQL deployments.

OrderFlow requires MariaDB database server version 5.5. The application has also been tested on the MariaDB 10.0 series.

The simplest configuration is where the database resides on the same machine as the OrderFlow application. In this case, install MariaDB using the MariaDB Repository Generator to find the correct repository for the operating system, for version 5.5.

Our convention is to use the server root password as the database root password, in which case this should be supplied during the installation process. If this convention is not followed, note the database server password used, and record it in *RemoteMan*.

Note that when other database users are created, their passwords will need to be changed to ensure that the database remains secure.

> **Note**
>
> Sometimes the database is required to reside on a dedicated database server - this is usually the case if a customer is hosting the environment themselves. In this case, the installation of MariaDb is not necessary.

OrderFlow holds PDF courier labels, imported data files and other large file files as database blobs. The database environment should be configured to ensure these can be included in database backups. In MySQL and MariaDB environments this can be done by setting the "max_allowed_packet" parameter to 1GB.

Apache Ant

Apache Ant is used to drive the deployment of OrderFlow. It currently needs to be at version 1.7.0 or greater. It can be installed using the following command:

```
sudo apt-get install ant
```

Curl

Curl is used pull files from the OrderFlow servers, both as part of the initial install and in subsequent upgrades. It can be installed using the following command:

```
sudo apt-get install curl
```

Zip

Zip is used compress and decompress files, it is used in the initial install and in subsequent upgrades. It can be installed using the following command:

```
sudo apt-get install zip
```

Debian Runlevel Configuration Tool

This tool configures system services in connection with system runlevels. It turns on & off services using the scripts in `/etc/init.d/`. We use it to allow configuration of init.d to enable OrderFlow to autostart after a reboot. Install using the following command:

```
sudo apt-get install rcconf
```

And run using the following command:

```
sudo rcconf
```

You will be presented with a screen that lists the executable files in `/etc/init.d/`, alongside a checkbox that shows whether they will be executed at start-up. Once an OrderFlow instance has been deployed, there will be an entry here for that instance, allowing it to be set to start up automatically, if required.

Note that if an error occurs when running `rcconf` reporting that "rcconf needs dialog or whiptail", then create the following symbolic link:

```
sudo ln -s /bin/whiptail /usr/bin/whiptail
```

Server Hardening

To enhance the server security, additional measures are taken. The following sections detail these measures; all examples assume the ubuntu operating system.

Admin User Password Change

As detailed in the Operating System section, the password for the `rtdadmin` user should be changed to a value that is recorded in *RemoteMan*. This can be done by using the following command (when logged in as `rtdadmin`):

```
sudo passwd
```

Move SSH Port

Moving the SSH port from 22 to 91 reduces the risk of the server being seen by casual scans. To do this, edit `/etc/ssh/sshd_config` and ensure that the `Port` key has the value `91`:

```
# What ports, IPs and protocols we listen for
Port 91
```

Firewall Configuration

SSH access should be restricted to port 91, and from only the main and backup proxy servers. This means that to SSH onto the OrderFlow server, a user will have to either do this from the proxy server, or "tunnel" through it.

We use ubuntu's Uncomplicated Firewall to configure this, by running the following commands:

```
sudo ufw allow from 188.65.36.243 to any port 91
sudo ufw allow from 164.177.149.218 to any port 91
sudo ufw allow 443
sudo ufw allow 8443
sudo ufw allow 9595
sudo ufw allow 9696
sudo ufw enable
```

The output from the last command will be something like:

```
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing)
New profiles: skip

To                         Action      From
--                         ------      ----
443                        ALLOW IN    Anywhere
8443                       ALLOW IN    Anywhere
9595                       ALLOW IN    Anywhere
9696                       ALLOW IN    Anywhere
91                         ALLOW IN    188.65.36.243
91                         ALLOW IN    164.177.149.218
443                        ALLOW IN    Anywhere (v6)
8443                       ALLOW IN    Anywhere (v6)
9595                       ALLOW IN    Anywhere (v6)
9696                       ALLOW IN    Anywhere (v6)
```

Database Password Changes

To ensure that the database remains secure, any users added to the database need a secure password. Database passwords can be changed by accessing the MariaDB client:

```
mysql -uroot -p
```

Then running the following SQL commands, for example:

```
UPDATE mysql.user SET password=PASSWORD('some decent password') WHERE user='root';
UPDATE mysql.user SET password=PASSWORD('some decent password') WHERE user='rtdadmin';
FLUSH PRIVILEGES;
```

OrderFlow Deployment

Assuming that an environment has been set up as previously described, OrderFlow instance(s) can be deployed by following the instructions in this section.

New OrderFlow Installation

Before OrderFlow can be deployed, there are some scripts and configuration files that need to be installed and edited. This process is described in the following section.

Bootstrapping Deployment Scripts

This process starts with downloading the "Deployment Scripts Bootstrap" Ant build file. This is normally done as the `rtdadmin` user in its home directory:

```
cd ~
mkdir deploy
cd deploy

curl -u build -o build.xml https://downloads.realtimedespatch.co.uk/general/build.xml
```

The password for `downloads.realtimedespatch.co.uk` is available in *RemoteMan*.

This will result in the following Ant build file:

```
/home/rtdadmin/deploy/build.xml
```

Once downloaded, run the following command to install the deployment environment (from the same directory):

```
ant install
```

This downloads and unzips the deployment scripts into the `deploy-host-jetty` subdirectory:

```
  ~/deploy$ ls -1 deploy-host-jetty/build*
  deploy-host-jetty/build.properties
  deploy-host-jetty/build.xml
  ~/deploy$
```

Note

In some circumstances, it may be necessary to manually install the certificate for `downloads.realtimedespatch.co.uk` , as follows:

`curl -u build -o InstallCert.jar https://downloads.realtimedespatch.co.uk/general/InstallCert.jar`

`java -cp InstallCert.jar InstallCert downloads.realtimedespatch.co.uk`

`java -cp InstallCert.jar InstallCert remote.realtimedespatch.co.uk`

DEPLOYMENT PROPERTIES

All the local configuration for the deployment of OrderFlow is held in the `deploy-host-jetty/build.properties` file. This file should be checked before every deployment command is invoked, to ensure that the properties within are correct.

A typical `build.properties` file looks like this:

```
download.dir=../downloads
download.user=********
download.password=********
download.fetch.base.url=https://downloads.realtimedespatch.co.uk

module.base.uri=https://remote.realtimedespatch.co.uk/web
module.user=********
module.password=********

#Uncomment for Linux
application.base.dir=/usr/share

#Uncomment for Windows
#application.base.dir=/RealtimeDespatch/

app.name=orderflow
customer.name=rtd
instance.name=test
```

Each property is expanded upon in more detail here:

Property Name	Example Value	Usage
download.dir	../downloads	The directory (relative to `deploy-host-jetty`) into which downloaded files are placed.
download.user	downuser	The (Basic Authentication) user name submitted to the `download.fetch.base.url` when downloading files.
download.password	downpass	The (Basic Authentication) password submitted to the `download.fetch.base.url` when downloading files.
download.fetch.base.url	\https://downloads.realtimedespatch.co.uk	The location from which to download core OrderFlow files.
module.base.uri	\https://remote.realtimedespatch.co.uk/web	The location from which to download optional module files.
module.user	moduser	The (RemoteMan) user name submitted to the `module.base.uri` when downloading files.
module.password	modpass	The (RemoteMan) password submitted to the `module.base.uri` when downloading files.
application.base.dir	/usr/share	The (absolute) directory under which OrderFlow is to be deployed.
app.name	orderflow	The application name of OrderFlow, usually left at its default value.
customer.name	rtd	The short (lower-case) customer name that the instance is for.
instance.name	test	The OrderFlow instance name, to distinguish it from other instances.

Note that the last three properties listed above are used to construct the database name, to which the OrderFlow instance will connect. For example, the example values above would result in the application attempting to connect to a database called `orderflow_rtd_test`.

DEPLOYMENT SCRIPTS UPDATE

If the deployment scripts need updating, for example, if a new version is available, then the following command (as `rtdadmin`, **not** via `sudo`):

```
ant update
```

This command backs up the `deploy-host-jetty/build.properties` file (which may have modified), replaces the deployment scripts folder with the latest from `downloads.realtimedespatch.co.uk`, then restores the backed up properties file.

Installation With Database

If the database is installed on the application server, then the OrderFlow application and database schema can both be installed at the same time.

To install OrderFlow *and* its database, navigate to `/home/rtdadmin/deploy/deploy-host-jetty/` and run the following command:

```
sudo ant install-with-db
```

Note that this may require `sudo` privilege as it creates directories and writes files to below the `application.base.dir`, which is usually owned by `root`.

The script will prompt for several parameters:

- **Target revision** - this will be the revision in the name of the built artefact, as found on the Downloads_ server, e.g. for the zip file called `orderflow-45561-jetty.zip`, revision `45561` should be entered.
- **Checkout path** - this is the path relative to \https://downloads.realtimedespatch.co.uk/orderflow/ in which the built artefact can be found. This depends on whether a tagged, branch or trunk build is being deployed. Examples include `tags/3.6.8-lowestoft_from_45560` or `branches/3.6.7-london_from_44661`. (If no checkout path is supplied then a *trunk* build is assumed.)
- **Database user** - the database user needs to have sufficient privilege to make schema changes on the database, so typically `rtdadmin` or sometimes `root` is used here. (Note that the OrderFlow application accesses the database using a less-privileged user, as defined in `config/jdbc.properties` found under the `application.base.dir`.)
- **Database password** - this is the password for the user previously supplied. The appropriate values for these parameters can be found in RemoteMan.

Note

The `install-with-db` installation target performs the following tasks:

- Asks for confirmation that the database should be installed. This requires you to type `true` and hit return, to prevent accidentally overwriting any existing database.
- Checks that the instance is not already installed.
- Creates a database, named from the values in `build.properties` accordingly.
- Creates a database admin user (called `rtdadmin` by default), and grants suitable privileges to this and the user that OrderFlow will use. (This is a shorter name derived from the database name - e.g. `ord_rtd_test`.)
- Downloads the core OrderFlow zip and database files (from under the `download.fetch.base.url`). These contain the core Java application and all necessary scripts, property files, commisioning data etc.
- Downloads any non-core module files available to the customer (from under the `module.base.uri`).
- Copies (and expands) the downloaded files and modules to the `application.base.dir`.
- Copies the expanded application start and `init.d` scripts to the application base and `/etc/init.d/` directories respectively.
- Copies configuration files to the `config` directory below the `application.base.dir`.
- Loads the downloaded commissioning data into the application's database, and rolls the database forward to the application's version.

One remaining installation task to perform is the installation of **database triggers**, which (until #9369 has been implemented) must be done manually, by running the following commands (with database name and username substituted accordingly):

```
cd /tmp
curl -u build -o audit_product_inventory.sql https://support.orderflow-wms.co.uk/svn/orderflow/trunk/rtd2/
database/triggers/audit_product_inventory.sql
curl -u build -o drop_all_triggers.sql https://support.orderflow-wms.co.uk/svn/orderflow/trunk/rtd2/database/
triggers/drop_all_triggers.sql
curl -u build -o report_configuration_audit_delete.sql https://support.orderflow-wms.co.uk/svn/orderflow/
trunk/rtd2/database/triggers/report_configuration_audit_delete.sql
curl -u build -o report_configuration_audit_update.sql https://support.orderflow-wms.co.uk/svn/orderflow/
trunk/rtd2/database/triggers/report_configuration_audit_update.sql
curl -u build -o report_design_audit_delete.sql https://support.orderflow-wms.co.uk/svn/orderflow/trunk/rtd2/
database/triggers/report_design_audit_delete.sql
curl -u build -o report_design_audit_update.sql https://support.orderflow-wms.co.uk/svn/orderflow/trunk/rtd2/
database/triggers/report_design_audit_update.sql
mysql -urtdadmin -p orderflow_rtd_test < drop_all_triggers.sql
mysql -urtdadmin -p orderflow_rtd_test < audit_product_inventory.sql
mysql -urtdadmin -p orderflow_rtd_test < report_configuration_audit_delete.sql
mysql -urtdadmin -p orderflow_rtd_test < report_configuration_audit_update.sql
mysql -urtdadmin -p orderflow_rtd_test < report_design_audit_delete.sql
mysql -urtdadmin -p orderflow_rtd_test < report_design_audit_update.sql
```

Installation Without Database

If database is on a remote system to which you don't have root access, then it might make sense to install the database separately.

To install OrderFlow (without installing the database), navigate to `/home/rtdadmin/deploy/deploy-host-jetty/` and run the following command:

```
sudo ant install-no-db
```

Note that this requires `sudo` privilege as it creates directories and writes files to below the `application.base.dir`, which is usually owned by `root`.

The script will prompt for several parameters:

- **Target revision** - this will be the revision in the name of the built artefact, as found on the Downloads_ server, e.g. for the zip file called `orderflow-45561-jetty.zip`, revision `45561` should be entered.
- **Checkout path** - this is the path relative to \https://downloads.realtimedespatch.co.uk/orderflow/ in which the built artefact can be found. This depends on whether a tagged, branch or trunk build is being deployed. Examples include `tags/3.6.8-lowestoft_from_45560` or `branches/3.6.7-london_from_44661`. (If no checkout path is supplied then a *trunk* build is assumed.)

Note

The `install-no-db` installation target performs the following tasks:

- Asks for confirmation that the database should not be installed.
- Checks that the instance is not already installed.
- Downloads the core OrderFlow zip and database files (from under the `download.fetch.base.url`). These contain the core Java application and all necessary scripts, property files, commisioning data etc.
- Downloads any non-core module files available to the customer (from under the `module.base.uri`).
- Copies (and expands) the downloaded files and modules to the `application.base.dir`.
- Copies the expanded application start and `init.d` scripts to the application base and `/etc/init.d/` directories respectively.
- Copies configuration files to the `config` directory below the `application.base.dir`.

The following commands can be run to install the database separately:

```
sudo ant db-create
sudo ant db-load
```

Note that after running the above scripts, the database triggers will still need to be installed as described in the previous section.

Installation Verification

The following command can be run to check that OrderFlow has been installed correctly:

```
ant check-environment
```

Starting OrderFlow

OrderFlow can be started in one of two ways. It can be started directly using the instance-specific `init.d` script, for example:

```
```

```
sudo /etc/init.d/orderflow_rtd_test start
```

Or the following Ant target can be invoked, from the `deploy-host-jetty` directory. If this method is used, the `build.properties` file **must** be re-checked beforehand, to ensure that the correct instance is being referenced:

```
sudo ant application-start
```

Verifying OrderFlow Start-up

The easiest way to verify that OrderFlow has started is by attempting to access its desktop interface from a browser, a minute or two after starting it. However, any problems experienced on start-up will not necessarily be apparent during an ad-hoc navigate around the interface.

Instead, OrderFlow's log files should be inspected during start-up, to verify that no problems have occurred.

The `logs` directory can be found under the application's base directory, for example in `/usr/share/orderflow/rtd/test/logs/`. Today's OrderFlow log is called `orderflow.log`, and today's request log file is called `accesslog.log`. The OrderFlow log file should be tailed during start-up and watched for any abnormal behaviour. This can be achieved (from anywhere on the file system) by running a command similar to the following:

```
  tail -f /usr/share/orderflow/rtd/test/logs/orderflow.log
```

"Abnormal behaviour" to look for would be any of the following:

- Stack traces - lengthy listings of qualified method names, with line numbers, that pinpoint where an error has occurred, e.g.:

```
  2017-03-22 16:30:37,025 ERROR [DefaultModuleRuntimeManager] Failed to handle loading of application module:
rtd2-batch
  org.springframework.beans.factory.xml.XmlBeanDefinitionStoreException: Line 359 in XML document from class
path resource [rtd2-batch-context.xml] is invalid; nested exception is org.xml.sax.SAXParseException;
lineNumber: 359; columnNumber: 9; cvc-complex-type.2.3: Element 'beans' cannot have character [children],
because the type's content type is element-only.
          at
org.springframework.beans.factory.xml.XmlBeanDefinitionReader.doLoadBeanDefinitions(XmlBeanDefinitionReader.java
399)
          at
org.springframework.beans.factory.xml.XmlBeanDefinitionReader.loadBeanDefinitions(XmlBeanDefinitionReader.java:
336)
          ...
          at org.mortbay.component.AbstractLifeCycle.start(AbstractLifeCycle.java:50)
          at RunJetty.main(RunJetty.java:113)
          at OrderFlow.main(OrderFlow.java:44)
  Caused by: org.xml.sax.SAXParseException; lineNumber: 359; columnNumber: 9; cvc-complex-type.2.3: Element
'beans' cannot have character [children], because the type's content type is element-only.
          at org.apache.xerces.util.ErrorHandlerWrapper.createSAXParseException(Unknown Source)
          at org.apache.xerces.util.ErrorHandlerWrapper.error(Unknown Source)
          ...
```

- Modules not loading - this can be detected by the presence of an error in the final list of modules, e.g.:

```
  2017-03-22 16:30:49,042 INFO [TransitionsLogger]
  ------------------------------------------------
  Module operations succeeded: false
  Number of operations: 132
    rtd2-api: UNLOADED_TO_LOADED
    rtd2-main: UNLOADED_TO_LOADED
    rtd2-hibernate: UNLOADED_TO_LOADED
    rtd2-dao: UNLOADED_TO_LOADED
```

```
    rtd2-process: UNLOADED_TO_LOADED
    rtd2-warehouse: UNLOADED_TO_LOADED
    rtd2-archive: UNLOADED_TO_LOADED
    rtd2-monitoring: UNLOADED_TO_LOADED
  ...
    rtd2-batch: UNLOADED_TO_LOADED
      org.springframework.beans.factory.xml.XmlBeanDefinitionStoreException: Line 359 in XML document from
class path resource [rtd2-batch-context.xml] is invalid; nested exception is org.xml.sax.SAXParseException;
lineNumber: 359; columnNumber: 9; cvc-complex-type.2.3: Element 'beans' cannot have character [children],
because the type's content type is element-only.
  ...
```

- Scheduler not started correctly (check that the following log is correct - use a live instance):

```
  2017-03-22 16:47:17,487 INFO [QuartzScheduler] Scheduler meta-data: Quartz Scheduler (v2.1.6)
'DefaultQuartzScheduler' with instanceId 'NON_CLUSTERED'
    Scheduler class: 'org.quartz.core.QuartzScheduler' - running locally.
    NOT STARTED.
    Currently in standby mode.
    Number of jobs executed: 0
    Using thread pool 'org.quartz.simpl.SimpleThreadPool' - with 10 threads.
    Using job-store 'org.quartz.simpl.RAMJobStore' - which does not support persistence. and is not clustered.
```

Stopping OrderFlow

If OrderFlow is required to be stopped for any reason, it can be done so in one of two ways (like start-up). It can be stopped directly using the instance-specific `init.d` script, for example:

```
sudo /etc/init.d/orderflow_rtd_test stop
```

Or the following Ant target can be invoked, from the `deploy-host-jetty` directory. If this method is used, the `build.properties` file **must** be re-checked beforehand, to ensure that the correct instance is being referenced:

```
sudo ant application-stop
```

Restarting OrderFlow Automatically

The server environment should be configured to start OrderFlow automatically whenever the server is rebooted. How to do this in the default Ubuntu server environment is covered in the section "Debian Runlevel Configuration Tool"

Note that OrderFlow should not be automatically restarted by external monitoring tools or utilities without this first having been discussed with the OrderFlow support team.

OrderFlow Upgrade

Existing OrderFlow instance(s) can be upgraded by following the instructions in this section.

Manual Steps

Before upgrading any instance of OrderFlow, it is essential to check if there are any *Manual Steps* to be performed, either before or after the main upgrade steps.

Manual steps are detailed on the following page:

https://support.orderflow-wms.co.uk/projects/matrixsupport/wiki/ManualSteps

To determine *which* manual steps apply to the current upgrade, the current version of OrderFlow needs to be determined. This can be found from the 'About' box in the running instance, or it can be found on the application server by inspecting the OrderFlow `current` symbolic link, as shown below:

```
rtdadmin@customertestnew:~$ ls -l /usr/share/orderflow/rtd/test/ | grep current
lrwxrwxrwx 1 root root   21 Dec 22  2014 current -> orderflow-35384-jetty
```

Any manual steps that reference revisions above the current version and below the target version must be read to see if they are applicable to the current instance being upgraded.

If applicable, the instructions must be followed. If it is not possible to follow the instructions for any reason, then a ticket should be raised and assigned as 'Awaiting Dev. Input' for actioning as soon as possible.

Upgrade With Database Changes

The most common form of OrderFlow upgrade is an upgrade to a tagged version. This includes database changes that correspond with the software changes, so these need to be applied to the database in the form of *roll-forward scripts*.

Disable monitoring before the upgrade: https://old.uptimerobot.com/login?rt=https://old.uptimerobot.com/dashboard.php Pause uptime robot

https://remote.realtimedespatch.co.uk/web/config/node/home.htm?menu=config/node Select the instance and click edit, then set the expected status as either blank or No Preference.

Perform a cleanup of old backups and orderflow folders: /usr/share/orderflow/*customer*/live/old : keep the last 3 files /home/rtdadmin/deploy/dbbackups: keep the last backup

Firstly, navigate to `/home/rtdadmin/deploy/deploy-host-jetty/` and check that the `build.properties` file references the correct instance. Then run the following command from this directory:

```
sudo ant upgrade-with-db
```

The script will prompt for several parameters:

- **Target revision** - this will be the revision in the name of the built artefact, as found on the Downloads server, e.g. for the zip file called `orderflow-45561-jetty.zip`, revision `45561` should be entered.
- **Checkout path** - this is the path relative to \https://downloads.realtimedespatch.co.uk/orderflow/ in which the built artefact can be found. This depends on whether a tagged, branch or trunk build is being deployed. Examples include `tags/3.6.8-lowestoft_from_45560` or `branches/3.6.7-london_from_44661`. (If no checkout path is supplied then a *trunk* build is assumed.)
- **Database user** - the database user needs to have sufficient privilege to make schema changes on the database, so typically `rtdadmin` or sometimes `root` is used here. (Note that the OrderFlow application accesses the database using a less-privileged user, as defined in `config/jdbc.properties` found under the `application.base.dir`.)
- **Database password** - this is the password for the user previously supplied. The appropriate values for these parameters can be found in RemoteMan.

Note

The `upgrade-with-db` target performs the following tasks:

- Asks for confirmation that the correct instance is to be upgraded.
  This depends upon properties set in the `build.properties` file.
- Stops the running instance, if required.
- Backs-up the database to `~rtdadmin/deploy/dbbbackups`, unless user directs otherwise. Note that it is **highly recommended** that the database is backed-up, as any bugs in roll-forward scripts have the potential to render the existing database un-usable!
- Compresses the database backup file.
- If the upgrade is to a *branch* version of OrderFlow, and database changes are being applied (which will be the case is the upgrade is from a previous tagged version of OrderFlow to a newer version that has already been branched), then the script asks for confirmation that this is correct.
- Downloads the core OrderFlow zip and database files (from under the `download.fetch.base.url`), for the target version concerned.
- Downloads any non-core version-specific module files available to the customer (from under the `module.base.uri`).
- Copies (and expands) the downloaded files and modules to the `application.base.dir`.
- Rolls the database forward to the application's version.

Once the upgrade is complete and any relevant manual steps have been applied, it is necessary to restart OrderFlow by using the following command:

```
sudo ant application-start
```

Verify that OrderFlow has started successfully by following the instructions in the Verifying OrderFlow Start-up section.

Note that some manual steps may be more easily applied once the application has been restarted.

Ensure you reenable the system monitoring.

Upgrade Without Database Changes

When the upgrade is within a **branch** of OrderFlow (i.e. between the branch's tag and a revision on that branch, or between two revisions on the same branch), the no database changes will be required.

Disable monitoring before the upgrade: https://old.uptimerobot.com/login?rt=https://old.uptimerobot.com/dashboard.php Pause uptime robot

https://remote.realtimedespatch.co.uk/web/config/node/home.htm?menu=config/node Select the instance and click edit, then set the expected status as either blank or No Preference.

Perform a cleanup of old backups and orderflow folders: /usr/share/orderflow/*customer*/live/old : keep the last 3 files /home/rtdadmin/deploy/dbbackups: keep the last backup

This kind of upgrade is quicker, as no database back-up is required. To upgrade with no database changes, navigate to `/home/rtdadmin/deploy/deploy-host-jetty/` and check that the `build.properties` file references the correct instance.
Then run the following command from this directory:

```
sudo ant upgrade-no-db
```

The script will prompt for the target revision and the checkout path (which will typically be a `branches` path).

Note

The `upgrade-no-db` target performs the following tasks:

- Asks for confirmation that the correct instance is to be upgraded.
  This depends upon properties set in the `build.properties` file.
- Stops the running instance, if required.
- Downloads the core OrderFlow zip and database files (from under the `download.fetch.base.url`), for the target version concerned.
- Downloads any non-core version-specific module files available to the customer (from under the `module.base.uri`).
- Copies (and expands) the downloaded files and modules to the `application.base.dir`.

Again, once the upgrade is complete and any relevant manual steps have been applied, OrderFlow needs to be started:

```
sudo ant application-start
```

After verifying that OrderFlow has started successfully, check that all relevant manual steps have been applied.

Ensure you reenable the system monitoring.

Environment Checklist

Software

- In addition to the OrderFlow environment the user 'rtdadmin' should have access to the 'ant','curl' and 'zip' utilties required during the initial deployment and subesquent upgrades.

Access

- The SSH user 'rtdadmin' should have access to the server (ideally on port 91)
- Incoming access for SSH user 'rtdadmin' should be restricted to the OrderFlow source IP addresses (see section "Operating System")
- The SSH user 'rtdadmin' should be given 'sudo' rights (or the permissions described in the section "Operating System")

- The database user 'rtdadmin' should have all rights to the OrderFlow database(s)
- Consider whether browser access to the OrderFlow environment should be restricted to fixed IP addresses

Backups, logs, data archiving and automatic startup

- The OrderFlow database should be included in a backup policy appropriate for mission critical business environments
- The OrderFlow log rotation functionality should be enabled to ensure log files do not continue to grow indefinitely
- Consider whether OrderFlow's data purge and data archive functionality can be used to remove data from the OrderFlow environment when it is no longer of value
- OrderFlow should start automatically when the server is powered up but should not be automatically restarted thereafter without discussion with the OrderFlow support team.

Commissioning Information

When commissioning a new system, the new installation will deliver a basic "vanilla" OrderFlow instance that contains no customer-specific data.

For OrderFlow to be usable, it needs to be configured according to the customer's needs, to allow them to use it to drive their warehouse operations. This configuration relies on extracting information from the customer - information such as location set-up, product catalogue, couriers in use, and so on.

This section details what information to extract and in what format. The following section details *how* this information can be used to configure OrderFlow appropriately.

Configuration

This section will be expanded upon in a future version.

Third-party Logistics (3PL) Organisations

This section will be expanded upon in a future version.