unipro

Project:

UUID Issue remediation

Title:

Release Guide / spin up guide (Draft)

Compiled by:	Dan Munn
Reference:	

Microsoft CERTIFIED Partner

1. Project Details

Project Name:	UUID Issue Remediation
Project No:	
Project Manager:	Deborah Dean
Our Project Reference:	UUID Issue Remediation
Customer Name:	Pfizer
Customer Project Reference:	
Customer Project Manager:	

2. Signoff

NAME	Signature	Date
Pfizer Engineering Lead	Matthew Saunders	
Programme Manager	Shaun Froome	
Lead Developer	Dan Munn	

3. Distribution

NAME	Copied (tick)	Date

4. Issue Log

Version	Description	Date	Author
1.0	Initial Draft	5/10/16	DM

5. Contents

1. Project Details....2
2. Signoff....2
3. Distribution....2
4. Issue Log....3
5. Contents....4
6. Introduction....5
7. Upgrade prerequisites....5
   7.1. Scheduling content freeze....5
   7.2. Identification of existing market version....5
   7.3. Providing a new backup of the environment....5
8. Version change requirements....6
9. Upgrading the market....7
   9.1. Update the Jenkins build configuration....7
   9.2. Execute a new build and monitor....7
   9.3. Validation of upgrade....7
10. Failed upgrades....7
   10.1. Jenkins modification....7
   10.2. Re-build the environment....8
   10.3. Database rollback....8

6. Introduction

This document outlines the steps for upgrading HCPP to the latest branch release (2.6.15 / 2.7.7 / 2.8.1) based on the current version of portal that the market is running.
Steps to execution:

1. Review and action the prerequisites
2. Review action the version change requirements
3. Review and action identified verification steps (note this will be agnostic to version).

Important Note: In the case of **ANY** failure within any step on an individual market, all versions should be rolled back to the version identified in the upgrade prerequisites.

Further note: The actions here should be performed on each market, and ensuring it is run on their dev3 / stage / production environments, some steps may reference this others will not, it is assumed that each step is run on each environment for each market. Deviation from this will cause delivery issues.

7. Upgrade prerequisites

There are number of actions that will need to be performed before an upgrade is safe to happen, this is to ensure that there is suitable opportunity to roll a market back in the case of failed upgrade.

Note: These actions will need to be performed on dev3, stage and production environments.

7.1. Scheduling content freeze

The Market will need to be contacted to indicate that their environments are not to be touched until the upgrade phase is complete, they will need to ensure that no user other than technical resources associated to the upgrade process is modifying the environment. The process cannot continue until the Market have acknowledged and actioned this freeze.

7.2. Identification of existing market version

It is important to identify for rollback activity (and upgrade path) accurate identification of current market HCPP version. Note this information will be referred to within this document as **source version**.

The best mechanism for quick identification of the current environments HCPP version is through the platform inspector, it can be quickly accessed through the Adhoc HCP Portal version report (found at https://pi.digitalpfizer.com/adhoc-hcp-portal-version).

7.3. Providing a new backup of the environment

Each environment that will be upgraded (dev3, stage, prod) will need a database backup, it would be prudent (although not absolutely required) to also backup the files on the environment that is being upgraded, as there could in some cases be alternative image styles that get rebuilt through point release upgrades.

In all cases database backups should be user-initiated backups and not the daily backups that are automated via Acquia.

8. Version change requirements

To ensure that the upgrade process is relatively smooth, there is a matrix provided that demonstrate based on source version what the target HCPP version number is and what the ideal tag for accessing this release would be.

SOURCE VERSION	TARGET VERSION	TARGET JENKINS TAG
2.6.0	2.6.15	7.x-2.6.x (STABLE)
2.6.1	2.6.15	7.x-2.6.x (STABLE)
2.6.2	2.6.15	7.x-2.6.x (STABLE)
2.6.3	2.6.15	7.x-2.6.x (STABLE)
2.6.4	2.6.15	7.x-2.6.x (STABLE)
2.6.5	2.6.15	7.x-2.6.x (STABLE)
2.6.6	2.6.15	7.x-2.6.x (STABLE)
2.6.7	2.6.15	7.x-2.6.x (STABLE)
2.6.8	2.6.15	7.x-2.6.x (STABLE)
2.6.9	2.6.15	7.x-2.6.x (STABLE)
2.6.10	2.6.15	7.x-2.6.x (STABLE)
2.6.11	2.6.15	7.x-2.6.x (STABLE)
2.6.12	2.6.15	7.x-2.6.x (STABLE)
2.6.13	2.6.15	7.x-2.6.x (STABLE)
2.6.14	2.6.15	7.x-2.6.x (STABLE)
2.7.0	2.7.7	7.x-2.7.x (STABLE)
2.7.1	2.7.7	7.x-2.7.x (STABLE)
2.7.2	2.7.7	7.x-2.7.x (STABLE)
2.7.3	2.7.7	7.x-2.7.x (STABLE)
2.7.4	2.7.7	7.x-2.7.x (STABLE)
2.7.5	2.7.7	7.x-2.7.x (STABLE)
2.7.6	2.7.7	7.x-2.7.x (STABLE)
2.7.7	2.8.1	7.x-2.8.x (STABLE)
2.7.8	2.8.1	7.x-2.8.x (STABLE)
2.7.9	2.8.1	7.x-2.8.x (STABLE)
2.7.10	2.8.1	7.x-2.8.x (STABLE)
2.7.11	2.8.1	7.x-2.8.x (STABLE)
2.8.0	2.8.1	7.x-2.8.x (STABLE)

It is an engineering task to ensure that the Jenkins tags are correct, if a tag is not available or identifiable, stop now and escalate this to engineering (or the Stabilisation team) immediately.

Look for the row from the matrix above which shows the **source version** identified in the pre-requisite steps (in the source version column), this row indicates the appropriate **target version** and **target Jenkins tag**; The identified target version and target Jenkins tag will be referred to by those names for the remainder of the document (with a view to referring to information identified in this step).

9. Upgrading the market

Providing that the steps above have been actioned, and that there is a clear source and target versions and also a target Jenkins tag then it is now the appropriate time to upgrade the market.

9.1. Update the Jenkins build configuration

Based on the **target Jenkins tag** identified earlier in the document, you should now access the environment's (note: for all dev3/stage/production) and ensure that the market environments are using the correct tag. In some cases (namely code only MTP) they are not able to use the stable tag, but instead should use the release there is an identifiable HCPP release based on the **target version**. Update the platform in line with the **target Jenkins tag** and then save the configuration.

9.2. Execute a new build and monitor

The next step is fairly simple, queue a new build for the environment that you are updating, upgrade steps within HCPP will automate the tasks for you.
In the above case, once the build has started it should be observed, it cannot be left for full un-observed automation. Courtesy of other engagements and environment tampering, there is the potential for error or race conditions which should be monitored for.

In the case of any observed severe irregularity with update steps (only) e.g. evidence of an infinite loop during the execution of the upgrade, and forceful errors; you should immediately jump to the failed upgrades section of this document.

9.3. Validation of upgrade

There are a number of internal tests that can be executed based on the post-upgrade environment data, these are identified in the test plan for the UUID upgrades and should be executed to correctly validate the upgrade.
Providing the validation was successful, the environment is considered upgraded and the "Failed upgrades" section below can be skipped (concluding the document). In the case of ANY failures the upgrade is to be considered failed and the "Failed upgrades" section becomes compulsory.

10. Failed upgrades

This section of the document only covers rollback and is not intended for execution unless there has been an issue identified in the steps above.

Note in the case of a single market environment failing, all of the market environments (dev3, stage, prod) should follow this guide and rollback to backups taken before the upgrade.

10.1. Jenkins modification

Modify the Jenkins environment configuration to use the HCPP build appropriate for **the source version** identified in the prerequisite steps e.g. 7.x-hcp-2.6.9 for a market who's **source version** is 2.6.9. Once the change has been completed save the configuration. If there is no tag available for the source version, then immediately consult the engineering team.

10.2. Re-build the environment

The environment needs to be re-built to prevent incomplete upgrades remaining within the environment, the build process may fail – however the code deployment steps are the passing requirement for this step.

10.3. Database rollback

Simply revert the database snapshot taken before the upgrade process.