


PROD to PPP Synchronization

Status: **IN PROGRESS**

Last review: March 2022 (sprint 37)

Documentation history:

- Page creation 15 Mar 2022 (sprint 37)

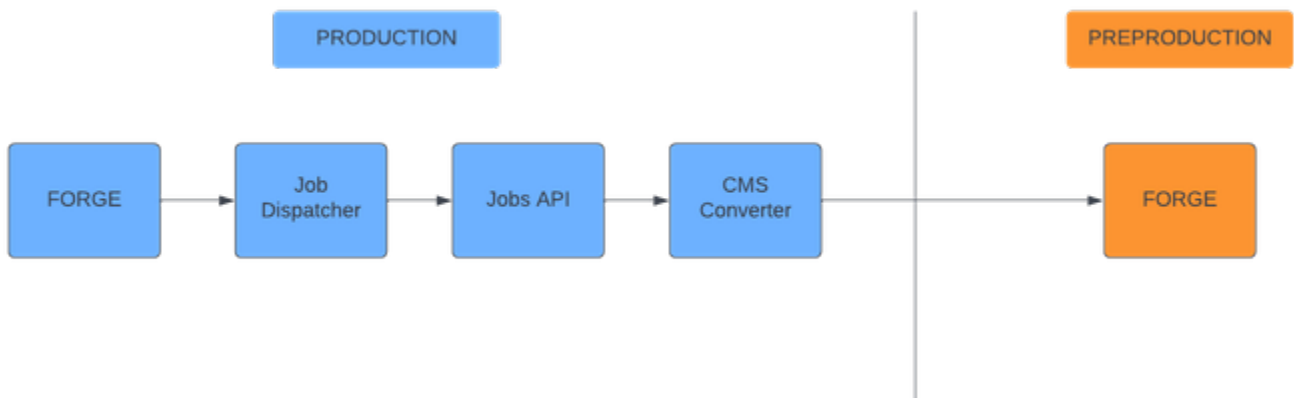
Reference  **B2P-6137** - Forge to Forge: synch between different environments **READY FOR REVIEW**

The aim of this page is to describe how Production and PreProduction environments could be put in sync constantly.

A similar process was in place last summer during the Tokyo2020 Games in order to synchronize Tokyo Forge with the <http://olympics.com>'s one, but that situation was adding one layer of complexity: the two Forges to put in sync were hosting different definitions of entities (e.g. custom body parts).

The current case is simpler, we are designing a synchronization between two instances of the same system hosting the same definitions of entities just on different environments.

Tokyo2020 Synchronization



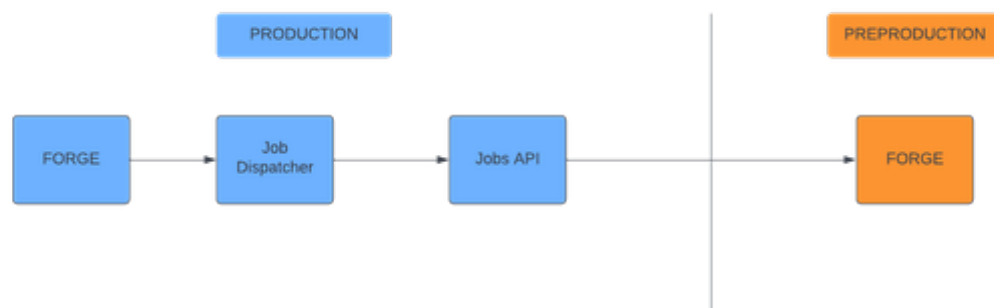
In the Tokyo2020 project, the requirement was to keep in sync with OC Forge some specific stories and VODs.

The specification of which story was supposed to be synched was managed via a custom field inside the Tokyo2020 Forge entity.

Technically speaking these were the needed steps to synchronize the stories:

1. When an editor was writing a story and wanted it to be pushed to OC Forge as well he had to select the specific field "Export To Olympicchannel.Com"
2. Once the story was published a notification was sent to the Service Bus (native behavior of Forge)
3. The Job Dispatcher (Deltatre product) was listening at the notification on the Bus
4. When the Job Dispatcher decoded a notification about a story with the specific sync flag it ran the specific Jobs API whose goal was to call the CMS Converter (part of the CMS importer product) in order to map the story to the target system (OC Forge)
5. CMS converter sent proper commands to the target Forge in order to recreate the entity resulting in a copy of the original one but the story parts were not supported by OC Forge.

Post Beijing2022 Synchronization



This scenario has already been put in place for other customers for other purposes, mainly to keep in sync different Forges in a multi-forge setup.

The same scenario could be applied to cover OCS needs putting in sync two environments.

As you can see the scenario is simpler than the one used for Tokyo2020 because in this case the two Forge is completely aligned in the definitions of entities so we can get rid of the CMS Converter from the diagram.

The flow is very close to the previous one with a direct connection between the Jobs API and the target Forge instance via the Deltatre ForgeSDK. Moreover, we are gonna remove the custom flag to decide which entity must be synched, because we want all the entities to be pushed to PPP.

1. Once an entity is published in PROD Forge a notification was sent to the PROD Service Bus (native behavior of Forge)
2. The PROD Job Dispatcher (Deltatre product) is listening to the notification on the Bus
3. When the PROD Job Dispatcher decodes a notification about an entity published it runs the specific PROD Jobs API
4. PROD Jobs API create the proper commands to be run on the PPP Forge via ForgeSDK
5. The entity is created/updated on the PPP Forge as a copy of the main one in PROD

Current Gaps

1. No Job Dispatcher is available in OCS infra so it must be deployed by Deltatre Devops. **No need for any Forge update** but two new services (Job Dispatcher and Jobs API) must be deployed on the Forge Kubernetes cluster with a couple of pods each per region as per availability of the service. About resource tuning, we cannot say now but we need to perform some tests once deployed in one environment.
2. Jobs API: as of now this part would require an integration of new code. All the custom entities managed by OCS should be analyzed and coded as a standard custom entity definition in order to let the code know how to map them and which commands must be sent to the PPP Forge. Moreover also the standard entities like stories must be reviewed in order to manage all the custom body parts used in the OCS Forge. The ball park figure for this activity is:
3. Just a note: this process is gonna synch the two systems from the moment we put in place the flow onwards, no old content will be synched because the trigger of the flow is the publication action on the master Forge. so before starting the live sync we should anyhow align the two systems manually once.

Job Dispatcher Notes

The setup of the job dispatcher is an activity that can be useful for a lot of other future improvements:

1. Migration on .NET Core also of the services, that can then be triggered by the Job dispatcher and avoid to use the current outdated approach of looking for events in MongoDB events collections or to listening directly to the RabbitMQ messages
2. We have an ongoing discussion also with SED team for push notification. The Job dispatcher could be the trigger for an endpoint used by SED team to send push notifications.

Estimations

Component	MD
Stories	5
CUSTOM Entities	3
Album	5
Photo	3

Tag	1
Documents	2
Editorial Sel.	2
Unpublish/Archive Logic	5
Devops to Setup Dispatcher	2
Integration tests	2
TOTAL	30