

|  |  |  |
| --- | --- | --- |
|  | Spécification US006 - Créer un ou des SIP pour le lot de versement  Version 1.0 – 21 octobre 2019 | |
| Gamme GED / AED | |  |

**REFERENCE DOCUMENT**

|  |  |  |
| --- | --- | --- |
| **Specification US006 DORY V0** | | |
| **Date de création : 21/10/2019** | **Date de révision :** | **Date d’approbation :** |
| **Auteur : R. DEVINANT** | **Vérifié par :** | **Approuvé par :** |

**HISTORIQUE DES MODIFICATIONS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Crée ou modifié le** | **Rédacteur** | **Nature des mises à jour et commentaires** |
| **1.0** | 21/10/2019 | R. DEVINANT | Initialisation |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Vérification du document** | | | |
| **Nom** | **Fonction** | **Date** | **Visa** |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Approbation du document** | | | | | |
| **Nom** | **Fonction** | **Date** | **Visa** |
|  |  |  |  |
|  |  |  |  |

SOMMAIRE

[REFERENCE DOCUMENT 2](#_Toc24097543)

[HISTORIQUE DES MODIFICATIONS 2](#_Toc24097544)

[SOMMAIRE 4](#_Toc24097545)

[Tables des figures 4](#_Toc24097546)

[1. Généralités 5](#_Toc24097547)

[1.1 Objectifs du Document 5](#_Toc24097548)

[2. User Story 0005 6](#_Toc24097549)

[2.1 Contexte fonctionnel 6](#_Toc24097550)

[2.2 Contexte technique 6](#_Toc24097551)

[2.2.1 Base de données 6](#_Toc24097552)

[2.2.2 Paramétrage 6](#_Toc24097553)

[2.2.3 Développement 7](#_Toc24097554)

Tables des figures

**Aucune entrée de table d'illustration n'a été trouvée.**

# Généralités

## Objectifs du Document

Ce document regroupe l’ensemble des principes techniques à mettre en œuvre dans le but de réaliser le développement de la User Story 006.

# User Story 0006

## Contexte fonctionnel

Dans le cadre de la V0 de DORY, notre client est le CNA. Le but est de permettre au système DORY de construire le nombre de SIP nécessaires à partir des documents et fichiers métadonnées d’un lot de versement.

## Contexte technique

Cette User Story correspond uniquement à du développement. Il est nécessaire de créer un nouvel agent dans le repository. Créer le nouvel agent dans le repository :

<http://gitlab.altair.recouv/dory/>

Sous le nom « doryAgentGenerationSip ». Ce projet doit être un projet Maven / Java.

Une fois les sources déployées sur le pc localement, il est nécessaire de créer le projet Eclipse (ou autre IDE Java mais en respectant les principes prévus dans la documentation d’installation du poste de développement). La documentation d’installation du poste de développement se trouve dans le répertoire suivant :

\\cer69fas1\REFERENTIEL\03- SI\AED\DORY\NOUVEL ARRIVANT\Documentation\01 - INSTALLATION POSTE DEVELOPPEMENT\01 - POSTE WINDOWS

### Base de données

Lors de l’appel du service REST permettant de demander la prise en charge de documents, des échanges avec la base sont effectués.

Les tables impactées sont les suivantes :

* do\_document
* st\_statut\_document
* lo\_log\_document
* hi\_historique\_statut\_document
* lo\_lot\_versement
* si\_sip
* st\_statut\_sip

Afin de comprendre et connaître le schéma de base de données, il est nécessaire de se référer aux documents présents dans le répertoire suivant :

[\\cer69fas1\REFERENTIEL\03- SI\AED\DORY\DOCUMENTATION\CONCEPTION\BDD DORY\](file:///\\cer69fas1\REFERENTIEL\03-%20SI\AED\DORY\DOCUMENTATION\CONCEPTION\BDD%20DORY\)

### Paramétrage

Aucun paramétrage spécifique n’est requis pour l’agent de scellement.

### Développement

#### Définition des logs de l’agent

Le principe est de créer un fichier tournant contenant toutes les actions effectuées par l’agent afin de proposer les traces relatives à DORY.

#### Description technique

L’agent de génération se SIP doit

Tâches planifiées

Exécuter par le planificateur de tâche

Mécanisme de planification de tâche (threadpoolscheduler)

Tâche exécutable à une date connue

La tâche sera toujours exécutée X temps entre la fin d’une exécution et le début de la suivante

Exécution d’une tâche périodiquement.

Délai entre

Planifié des tâches exécutable

CronTrigger est utilisé pour planifier une tâche en focntion d’une espression cron

Exécution périodique de tâche en fonction d’une expression cron.

Planifier une tâche exécutable à l’aide du support technique de Spring

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Le rôle de cet agent est de générer périodiquement des SIP à partir des données provenant de la base de données.

Spring nous fournit depuis la version 3.0, un mécanisme de planification des tâches. Un support technique permettant de planifier l’exécution d’une ou plusieurs tâches.

Tout ce que nous avons à faire est de créer une tâche (le générateur de SIP) exécutable pour la planification (*ThreadPoolTaskScheduler*), puis de choisir une stratégie de planification appropriée (planification avec cronTrigger).

Un bean *threadPoolTaskScheduler* configuréest créé dans la classe *ThreadPoolTaskSchedulerConfig*.

@Bean

public ThreadPoolTaskScheduler threadPoolTaskScheduler(){…}

Il sera utilisé par notre programme (Agent de génération de SIP) pour gérer les exécutions asynchrones des tâches par des threads. Notez que lors de la configuration de ce bean, un pool de threads peut être créer. Mais dans notre cas, nous n’allons créer qu’un seul thread pour éviter que la même tâche soit exécutée simultanément par plusieurs threads. Le but est d’avoir une exécution synchrone des tâches. Vu qu’ici, nous n’avons qu’une seule tâche qui sera exécutée périodiquement, il est préférable qu’on attende la fin de l’exécution de la tâche courante avant de passer à une autre.

threadPoolTaskScheduler.setPoolSize(1);

Comme mentionner ci-dessus, nous n’avons qu’une seule tâche qui sera exécutée périodiquement par un thread :

1. En utilisant les services fournis par la librairie database-acces, nous allons rechercher tous les lots de versements au statut « HORODATE »
   1. Pour chaque lot de versement au statut « HORODATE », nous allons rechercher (en utilisant la librairie database-access) tous les documents au statut « HORODATE » de ce lot de versement.
2. Pour chaque lot de versement au statut « HORODATE »
   1. Nous allons rechercher (en utilisant la librairie database-access) par paquet de X (X étant le nombre de documents maximum permis pour la construction d’un SIP) les documents au statut « HORODATE » du lot de versement courant.
   2. Pour chaque paquet de documents du lot de versement courant, nous allons enlever tous ceux qui ne remplissent pas toutes les conditions nécessaires pour être intégrer dans un SIP.

Les conditions sont les suivantes :

* Présence des métadonnées obligatoires : Code RND et l’id d’archivage.
* Le fichier binaire associé au document est présent physiquement.
* Le fichier des métadonnées est présent physiquement.
  1. Le statut des documents qui ne remplissent pas toutes les conditions nécessaires pour être intégrer dans un SIP sera changer en « ERREUR ». Et le statut des autres documents valide sera changer en « CREATION\_SIP ». Notons que nous traçons tout changement de statut d’un document dans la table hi\_historique\_statut\_document en utilisant la librairie database-access.
  2. On va créer un SIP à partir des documents au statut « CREATION\_SIP » du paquet courant.
  3. Les documents qui sont bien intégrés dans le SIP passent le statut à « INTEGE\_SIP ». Alors que les autres passent le statut à « ERREUR ».
  4. Création d’un enregistrement dans la table SIP et ajouter les associations entre le SIP qui vient d’être créer avec les documents et le lot courant.

1. Une fois l’horodatage effectué, mettre à jour le statut du document à « HORODATE »
   1. Dans le cas où le document scellé correspond au dernier d’un lot et donc que l’ensemble des documents d’un lot sont scellés passer le statut du lot à « HORODATE »

L’idée est d’avoir les fils toujours existants, de sorte que nous n’aurons pas à payer les frais généraux de temps pour les créer à chaque fois

Notre planificateur et gestionnaire de tâche est le bean *ThreadPoolTaskScheduler.* Ce bean

Spring nous fournit la classe *ThreadPoolTaskScheduler* pour gérer et planifié les tâches à exécuter.

Dans la classe *ThreadPoolTaskSchedulerConfig*, nous avons configuré un bean *threadPoolTaskScheduler*. Ce bean à été configurer de manière qu’il exécute des tâches de manière asynchrone en fonction de la taille du pool configuré.

La tâche en question dans ce gé

Un nombre fixe de thread est alloué pour exécuter les tâches planifiées.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Dépuis la version 3.0 de Spring, il nous o

Le but est de créer une tâche planifiée et qui sera exécutée périodiquement.

Spring nous fourni un support technique permettant de planifier une tâche exécutable.

Spring nous fourni un mécanisme de planification de tâche (TaskScheduler).

Ainsi

L’agent de d’horodatage doit repérer les documents à horodater, effectuer le scellement et mettre à jour les statuts de lots et documents.

1. Rechercher dans la table des documents les documents à horodater
   1. Les documents concernés sont ceux au statut « SCELLE »
2. Effectuer l’horodatage par un appel du WS du centre de service de signature
   1. Exemple d’URL : <http://cnp69devdxsapp1.cer69.recouv:24943>
   2. Le retour de l’appel du WS doit être stockée dans la table document
3. Une fois l’horodatage effectué, mettre à jour le statut du document à « HORODATE »
   1. Dans le cas où le document scellé correspond au dernier d’un lot et donc que l’ensemble des documents d’un lot sont scellés passer le statut du lot à « HORODATE »