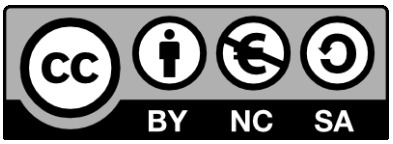
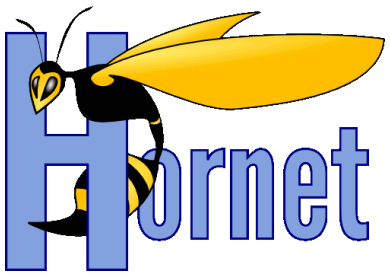


[HORNET]
Migration d'un projet Hornet 3.6 vers Hornet 3.10



Développement Hornet 3.10

Version : 1.0 du 05/01/2015 - **Etat** : Validé

SUIVI DES MODIFICATIONS

Version	Auteur	Description	Vérification	Date
1.0	O. Rousseil	Mise à jour de Hornet Version 3.6 vers Hornet Version 3.10		

DOCUMENTS DE REFERENCE

Titre
HORNET_GUI_Création d'un projet Hornet 3.10
HORNET_GUI_Guide du développeur Hornet 3.10
HORNET_GUI_Guide de paramétrage_1.4
HORNET_NOR_Normes d'encodage_1.0

SOMMAIRE

TABLEAUX

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

FIGURES

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

1 Objectifs du document

Dans le cadre de la mise en place des évolutions au sein du Framework Hornet, ce document spécifie les actions à effectuer au sein d'un projet de type Hornet pour migrer de la version 3.6 de Hornet vers la nouvelle version du Framework Hornet 3.10.

1.1 Principales évolutions

Cette nouvelle version 3.10 d'Hornet comporte principalement les évolutions et corrections suivantes par rapport à la version 3.6 :

- Mise à jour de l'utilisation de quartz
- Préconisation d'utilisation des mails
- Listener pour la journalisation d'évènements.
- Intégration de la librairie EHCache
- Ajout de la librairie jfreechart
- AppliTutoriel publiable sur adullact
- Ajout de metrologiefilter + Spring AOP
- Externalisation des javascripts
- Réorganisation des fichiers de configuration Spring
- Migration HttpClient 3 vers HttpComponents 4
- Evolution vers Spring Security 3.2.5
- Evolution vers Spring 3.2.11
- Migration vers Quartz 2.2.X
- Extension du composant Rattachement
- Classes de Pagination transférées de hornettemplate vers hornetserver
- Mise à jour de Tiles Request en version 1.0.5
- Amélioration de la qualimétrie (principalement sur hornetserver)
- Divers corrections de bug technique et sur l'ergonomie
- Factorisation des notifications

1.2 Versions cible des composants

Ce document est basé sur les versions de composants suivantes :

- ✓ hornetserver 3.10.2
- ✓ hornetclient 3.10.2
- ✓ hornettemplate 3.10.2

1.3 Mode d'emplois

La migration doit se faire dans l'ordre des chapitres qui sont structuré de la même manière :

- Un rappel et/ou préambule conditionne la réalisation du chapitre.
- Une partie « Tâche à réaliser » décrit la migration unitaire. Un tableau montre les fichiers à modifier dans la version 3.6 et dans la nouvelle version. Les modifications sont surlignées en jaune. Si le nom du fichier est connu, il est noté dans le tableau.
- Une dernière partie permet de vérifier la bonne migration.

2 Mise à jour technique

2.1 Configuration du projet sous Eclipse

2.1.1 Prérequis

Avant de commencer cette migration, il faut avoir installé et configuré l'environnement de développement conformément au guide de paramétrage, en particulier pour la configuration de Tomcat 6 et JDK 6 dans Eclipse.

2.2 Configuration de l'environnement de l'application à migrer

2.2.1 Migration des librairies clientes : Hornet client

Les thèmes et JavaScript doivent être installés sur un serveur de framework Hornet.

2.2.1.1 Rappel

Le framework Hornet est constitué de plusieurs parties :

- hornetserver : Fichiers de configuration ou fichiers Java, ils sont utilisés sur le serveur web.
- hornetclient : Fichiers JavaScript ou CSS (thème defaut, diplonet et francediplo), ils sont utilisés côté client (sur le navigateur)

2.2.1.2 Prérequis

Vous devez posséder les fichiers des librairies à mettre à jour:

- Hornet 3.10.2
- YUI 3.17.2,
- YUI Gallery 2014.02.13-03-13.

Si vous ne les possédez pas, le chapitre suivant décrit comment les obtenir.

2.2.1.2.1 Récupération des sources du framework Hornet à partir de hornettemplate

La migration vers Hornet 3.10.2 nécessite de récupérer de nouveaux fichiers et des fichiers mis à jour dans cette nouvelle version du framework. Ainsi, la source de ces fichiers doit être un projet « modèle » créé à partir de hornettemplate (cf. Guide de création d'un projet Hornet).

Dans la suite de ce document, sauf mention d'une autre source, les fichiers mentionnés sont donc ceux du projet créé à partir de hornettemplate **3.10.2.**

2.3 Tâche à réaliser : mise à jour des librairies

2.3.1 Mise à jour des ressources statiques : hornetclient

- Ajouter les nouvelles versions sous apache :

Fonctionnalité	Code Hornet 3.6	Code Hornet 3.10
Répertoire sous apache		
Ajouter une nouvelle version de Hornet javascript	`htdocs\hornetclient\3.6.2\fwk\**`	Ajouter la nouvelle version `htdocs\hornetclient\3.10.2\fwk\**`

Ajouter une nouvelle version de Hornet styles CSS	htdocs\hornetclient\3.6.2\themes\hornet-skin-defaut** htdocs\hornetclient\3.6.2\themes\hornet-skin-diplonet-3.6.2** htdocs\hornetclient\3.6.2\themes\hornet-skin-francediplo-3.6.2** ...	Ajouter les nouvelles versions htdocs\hornetclient\3.10.2\themes\hornet-skin-defaut** htdocs\hornetclient\3.10.2\themes\hornet-skin-defaut-xcombine** htdocs\hornetclient\3.10.2\themes\hornet-skin-diplonet-3.10.2** htdocs\hornetclient\3.10.2\themes\hornet-skin-diplonet-xcombine-3.10.2** htdocs\hornetclient\3.10.2\themes\hornet-skin-francediplo3.10.2** htdocs\hornetclient\3.10.2\themes\hornet-skin-francediplo-xcombine-3.10.2**

- Relancer Apache httpd

2.3.1.1 *Vérification*

- Lancer Apache httpd.
- A l'aide d'un navigateur internet, accéder à un des fichiers des librairies précédemment déposées sous apache.

*Exemple : [http://localhost/hornet]/hornetclient/*3.10.2*/fwk/hornetconfig/hornetconfig-min.js*

2.3.2 Migration des librairies (jar) serveur : hornetserver

A partir d'une tâche ANT, mise à jour des fichiers jar Hornet présents dans le dossier lib de l'application.

2.3.2.1 *Prérequis*

Vous devez posséder les fichiers des librairies à mettre à jour :

- Hornet Serveur 3.10.2 (« hornetserver-core » et « hornetserver-web »)
- Autres librairies hornetserver (hornetserver-httpparam, hornetserver-typemime, …)

Si vous ne les possédez pas, se reporter au chapitre: Récupération des sources du framework Hornet à partir de hornettemplate

2.3.2.2 *Tâche à réaliser : gestion des dépendances*

Vous pouvez procéder de deux manières :

- ✓ Choix 1 : Exécuter la tâche Ant.
- ✓ Choix 2 : Remplacer les anciennes librairies Hornet par leur version 3.10.2

- De toutes manières, faire pointer le projet « **hornetserver-web**» et les autres librairies Hornet vers la version **3.10.2** :

Fonctionnalité	Code Hornet 3.6	Code Hornet 3.10
	Fichier ivy.xml	

Nouvelle version de hornetserver-web	`<dependency`
	`org="fr.gouv.diplomatie.hornet"`
	`name="hornetserver-web" rev="3.6.2"`
	`conf="compile->master,libraries;provided;runtime->master,libraries; test; sonar; cobertura"`
	`transitive="true">`
	[…]
	`</dependency>`
		`<dependency`
		`org="fr.gouv.diplomatie.hornet"`
		`name="hornetserver-web" rev="3.10.2"`
		`conf="compile->master,libraries;provided;runtime->master,libraries; test; sonar; cobertura"`
		`transitive="true">`
		[…]
		`</dependency>`

➲ Si des librairies spécifiques sont présentes pour votre projet dans WEB-INF/lib, vous devrez les recopier

- Choix 1 : Lancer la tâche Ant « init-dev » afin de récupérer les nouvelles librairies dans le répertoire : « WEB-INF/lib ».
- Choix 2 : Remplacer les librairies : « WEB-INF/lib». Les anciennes doivent, bien sûr, être supprimées.

2.3.2.3 Vérification

Vérifier la présence et la mise à jour des nouvelles dépendances à la suite du build (choix1):

- **hornetserver-core-3.10.2.jar**
- **hornetserver-web-3.10.2.jar**

Les anciens jar de Hornet en version 3.6 ne doivent plus être présents**.**

2.3.3 Prise en compte des nouvelles versions : hornetclient, et les thèmes CSS

2.3.3.1 Rappel

La version de la librairie YUI Gallery précédemment mise à jour sous le serveur apache, est définie dans les sources d'Hornet client (fichier config.js, variable GALLERY_VERSION)

2.3.3.2 Tâche à réaliser : prise en compte nouvelle version

- Modifier le fichier « envconfig/hornet.properties », afin de pointer les variables d'environnement pour pointer vers les nouvelles versions des librairies Hornet.

Fonctionnalité	Code Hornet 3.6	Code Hornet 3.10
Fichier envconfig/hornet.properties		

Nouvelle version de hornetserver-web	`# Éléments de configuration du framework`
	`fwkRoot= http://<URL SERVEUR>/hornetclient/3.6.2/fwk`
	`yui3Root= http://<URL SERVEUR>/yui/yui/3.17.2`
	`themeName=diplonet`
	`themeVersion=3.6.2`
	`combineType=normal`
	`debugHornetClient=false`
		`# Éléments de configuration du framework`
		`fwkRoot= http://<URL SERVEUR>/hornetclient/3.10.2/fwk`
		`yui3Root= http://<URL SERVEUR>/yui/yui/3.17.2`
		`themeName=diplonet`
		`themeVersion=3.10.2`
		`combineType=normal`
		`debugHornetClient=false`

2.3.3.3 Vérification

- Lancer votre application Web.
- Ouvrir la page d'accueil à l'aide de firebug (Firefox) ou les outils de développement sous chrome avec l'option réseau d'activé. Les fichiers 'css' ou 'js' chargés doivent être dans la bonne version

  > 3.17.2 pour YUI
  > 3.10.2 pour des fichiers Hornet

2.4 Mise à jour pour les tests de vie

Les anciennes bibliothèques des tests de vie, à savoir :

- testvie-1.0.jar
- testvie-encode-1.1.jar

sont remplacées par le jar **« hornetserver-testvie-3.10.2.jar »**.

Procéder au changement en modifiant votre fichier ivy.xml. La déclaration de la nouvelle dépendance doit être du type :

```
<dependency org="fr.gouv.diplomatie.hornet" name="hornetserver-testvie"
       rev="3.10.2" conf="compile->default" transitive="false">
       <artifact name="hornetserver-testvie" type="jar" ext="jar" />
</dependency>
```

Le nom de package des classes pour les tests de vie à également changé. Il convient de remplacer dans toutes vos sources les références ou appels aux classes **« fr.gouv.diplomatie.testvie.* »** par **« hornet.framework.testvie.* »**.

De même les appels en ligne de commande du type :

```
java –jar testvie-encode-1.0.jar test.xml spring-appContext-testvie-encode.xml
```

sont à remplacer par :

```
java –jar hornetserver-testvie-3.10.2.jar test.xml spring-appContext-testvie-encode.xml
```

2.5 Migration HttpClient 3 vers HttpComponents 4

Ci-joint un tableau récapitulatif des éléments à adapter pour migrer de HttpClient 3 vers HttpComponents 4.

	Commons HttpClient 3.x	HttpComponents HttpClient 4.x
Import	import org.apache.commons.httpclient.Credentials; import org.apache.commons.httpclient.Header; import org.apache.commons.httpclient.HostConfiguration; import org.apache.commons.httpclient.HttpClient; import org.apache.commons.httpclient. MultiThreadedHttpConnectionManager; import org.apache.commons.httpclient. UsernamePasswordCredentials; import org.apache.commons.httpclient.auth.AuthScope; import org.apache.commons.httpclient.methods.DeleteMethod; import org.apache.commons.httpclient.methods.GetMethod; import org.apache.commons.httpclient.methods.PostMethod;	import org.apache.http.Header; import org.apache.http.HttpEntity; import org.apache.http.HttpHost; import org.apache.http.HttpResponse; import org.apache.http.auth.AuthScope; import org.apache.http.auth.Credentials; import org.apache.http.auth.UsernamePasswordCredentials; import org.apache.http.client.HttpClient; import org.apache.http.client.methods.HttpGet; import org.apache.http.client.methods.HttpPost; import org.apache.http.conn.params.ConnRoutePNames; import org.apache.http.entity.ContentType; import org.apache.http.entity.StringEntity; import org.apache.http.impl.client.DefaultHttpClient; import org.apache.http.util.EntityUtils;
HttpClient	HttpClient client = new HttpClient(connectionManager);	DefaultHttpClient client = new DefaultHttpClient();

Proxy Configuration	client.getHostConfiguration().setProxy(proxyHost, proxyPort);
	client.getState().setAuthenticationPreemptive(true);
	Credentials cred = new UsernamePasswordCredentials(proxyUser, proxyPassword);
	client.getState().setProxyCredentials(AuthScope.ANY_HOST, proxyHost, cred)
		client.getCredentialsProvider(). setCredentials(
		new AuthScope(proxyHost, proxyPort),
		new UsernamePasswordCredentials(proxyUser, proxyPassword));
		HttpHost proxy = new HttpHost(proxyHost proxyPort);
		client.getParams().setParameter(ConnRoutePNames.DEFAULT_PROXY, proxy);

POST	PostMethod post = new PostMethod(); post.setPath(url.getFile());	HttpPost post = new HttpPost(url.getFile());
POST Header	post.setRequestHeader(key, (String) headers.get(key))	post.addHeader(key, (String) headers.get(key));
POST Body	post.setRequestBody(message);	StringEntity messageEntity = new StringEntity(message, ContentType.create("text/plain", "UTF-8")); post.setEntity(messageEntity);
Execute POST	client.executeMethod(post);	HttpHost targetHost = new HttpHost(url.getHost(), url.getPort(), url.getProtocol()); HttpResponse httpResponse = client.execute(targetHost, post);
Response Header	Header[] rspHeaders = post.getResponseHeaders();	Header[] rspHeaders = httpResponse.getAllHeaders();

Response Body	String responseMsg = post.getResponseBodyAsString()	StringBuffer buffer = new StringBuffer();
		BufferedReader reader = new BufferedReader(new InputStreamReader(httpResponse.getEntity().getContent()));
		String dataLine = null;
		while((dataLine = reader.readLine()) != null){
		buffer.append(dataLine);
		}
		String responseMsg = buffer.toString();

GET	GetMethod getMethod = new GetMethod(); getMethod.setPath(url.getFile());	HttpGet httpGet = new HttpGet(url.getFile());
GET Header	getMethod.setRequestHeader(key, value);	httpGet.addHeader(key, value);
Execute GET	client.executeMethod(getMethod);	HttpHost targetHost = new HttpHost(url.getHost(), url.getPort(), url.getProtocol()); HttpResponse httpResponse = client.execute(targetHost, httpGet);

2.6 Migration Quartz 1.x vers 2.x

2.6.1 Quartz Configuration

Les propriétés définis dans le fichier de configuration « quartz.properties » demeurent fonctionnelles et inchangées.

2.6.2 Scheduler API

	Quartz 1.x	Quartz 2.x
Typed Collections	String[] triggerGroups = sched.getTriggerGroupNames();	List<String> triggerGroups = sched.getTriggerGroupNames();

Job and Trigger Identification (Keys)	String[] triggersInGroup = sched.getTriggerNames("myGroup");
	Trigger trg = sched.getTrigger("myTriggerName", "myGroup");
	sched.unscheduleJob("myOtherTriggerName", null);
	trg.getName();
	trg.getJobName();
	JobDetail myJob = sched.getJobDetail("myJobName", "myGroup");
		import static org.quartz.TriggerKey.*;
		import static org.quartz.JobKey.*;
		...
		List<TriggerKey> triggersInGroup = sched.getTriggerKeys("myGroup");
		Trigger trg = sched.getTrigger(triggerKey("myTriggerName", "myGroup"));
		sched.unscheduleJob(triggerKey("myOtherTriggerName"));
		trg.getKey().getName();
		trg.getJobKey().getName();
		JobDetail myJob = sched.getJobDetail(jobKey("myJobName", "myGroup"));

Constructing Jobs and Triggers	JobDetail job = new JobDetail("myJob", "myGroup");
	job.setJobClass(MyJobClass.class);
	job.setDurability(true);
	job.setRequestsRecovery(true);
	job.getJobDataMap().put("someKey", "someValue");
	
	SimpleTrigger trg = new SimpleTrigger("myTrigger", null);
	trg.setStartTime(new Date(System.currentTimeMillis() + 10000L));
	trg.setPriority(6);
	trg.setJobName("myJob");
	trg.setJobGroup("myGroup");
	trg.setRepeatCount(SimpleTrigger.REPEAT_INDEFINITELY);
	trg.setRepeatInterval(30000L)
		import static org.quartz.TriggerBuilder.*;
		import static org.quartz.JobBuilder.*;
		import static org.quartz.DateBuilder.*;
		import static org.quartz.SimpleScheduleBuilder.*;
		...
		JobDetail job = newJob(MyJobClass.class)
		.withIdentity("myJob", "myGroup")
		.storeDurably()
		.requestRecovery()
		.usingJobData("someKey", "someValue")
		.build();
		
		Trigger trg = newTrigger()
		.withIdentity("myTrigger")
		.startAt(futureDate(10, IntervalUnit.SECONDS))
		.withPriority(6)
		.forJob(job)
		.withSchedule(simpleSchedule()
		.withIntervalInSeconds(30)
		.repeatForever())
		.build();

Changes Related To Listeners (JobListener, triggerListener, SchedulerListener)	scheduler.addGlobalJobListener(myGlobalJobListener);
	scheduler.addJobListener(myJobListener);
	...
	job.addJobListener(myJobListener.getName());
	...
		import static org.quartz.impl.matchers.GroupMatcher.*;
		...
		// no matcher == match all jobs
		scheduler.getListenerManager().addJobListener(myGlobalJobListener);
		// match (listen to) all jobs in given group
		scheduler.getListenerManager().addJobListener(myJobListener, jobGroupEquals("foo"));
		...

Changes Related To TriggersUtils	Date startDate = TriggerUtils.getEvenHourDate(new Date()); // next hour, straight up
	Trigger t = TriggerUtils.makeDailyTrigger(10,45); // every day at 10:45
	t.setStartTime(startDate);
		import static org.quartz.DateBuilder.*;
		import static org.quartz.TriggerBuilder.*;
		import static org.quartz.CronScheduleBuilder.*;
		...
		Trigger t = newTrigger()
		.withSchedule(cronScheduleDaily(10,45)) // every day at 10:45
		.startAt(evenHourDate(new Date()) / next hour, straight up
		.build();

Changes Related To DateIntervalTrigger	DateIntervalTrigger	CalendarIntervalTrigger

Changes Related To NthIncludedDay Trigger	NthIncludedDayTrigger	NthIncludedDayTrigger

2.6.1 Database schema changes for setups using JDBCJobStore

```
--
- drop tables that are no longer used
-
drop table qrtz_job_listeners;
drop table qrtz_trigger_listeners;
-
- drop columns that are no longer used
-
alter table qrtz_job_details drop column is_volatile;
alter table qrtz_triggers drop column is_volatile;
alter table qrtz_fired_triggers drop column is_volatile;
-
- add new columns that replace the 'is_stateful' column
-
alter table qrtz_job_details add column is_nonconcurrent bool;
alter table qrtz_job_details add column is_update_data bool;
update qrtz_job_details set is_nonconcurrent = is_stateful;
update qrtz_job_details set is_update_data = is_stateful;
alter table qrtz_job_details drop column is_stateful;
alter table qrtz_fired_triggers add column is_nonconcurrent bool;
alter table qrtz_fired_triggers add column is_update_data bool;
update qrtz_fired_triggers set is_nonconcurrent = is_stateful;
update qrtz_fired_triggers set is_update_data = is_stateful;
alter table qrtz_fired_triggers drop column is_stateful;
-
- add new 'sched_name' column to all tables
-
alter table qrtz_blob_triggers add column sched_name varchar(120) not null DEFAULT 'TestScheduler';
alter table qrtz_calendars add column sched_name varchar(120) not null DEFAULT 'TestScheduler';
alter table qrtz_cron_triggers add column sched_name varchar(120) not null DEFAULT 'TestScheduler';
alter table qrtz_fired_triggers add column sched_name varchar(120) not null DEFAULT 'TestScheduler';
alter table qrtz_job_details add column sched_name varchar(120) not null DEFAULT 'TestScheduler';
alter table qrtz_locks add column sched_name varchar(120) not null DEFAULT 'TestScheduler';
alter table qrtz_paused_trigger_grps add column sched_name varchar(120) not null DEFAULT 'TestScheduler';
alter table qrtz_scheduler_state add column sched_name varchar(120) not null DEFAULT 'TestScheduler';
alter table qrtz_simple_triggers add column sched_name varchar(120) not null DEFAULT 'TestScheduler';
alter table qrtz_triggers add column sched_name varchar(120) not null DEFAULT 'TestScheduler';
-
- drop all primary and foreign key constraints, so that we can define new ones
-
alter table qrtz_triggers drop constraint qrtz_triggers_job_name_fkey;
alter table qrtz_blob_triggers drop constraint qrtz_blob_triggers_pkey;
alter table qrtz_blob_triggers drop constraint qrtz_blob_triggers_trigger_name_fkey;
alter table qrtz_simple_triggers drop constraint qrtz_simple_triggers_pkey;
alter table qrtz_simple_triggers drop constraint qrtz_simple_triggers_trigger_name_fkey;
alter table qrtz_cron_triggers drop constraint qrtz_cron_triggers_pkey;
alter table qrtz_cron_triggers drop constraint qrtz_cron_triggers_trigger_name_fkey;
alter table qrtz_job_details drop constraint qrtz_job_details_pkey;
alter table qrtz_job_details add primary key (sched_name, job_name, job_group);
alter table qrtz_triggers drop constraint qrtz_triggers_pkey;
-
- add all primary and foreign key constraints, based on new columns
-
alter table qrtz_triggers add primary key (sched_name, trigger_name, trigger_group);
alter table qrtz_triggers add foreign key (sched_name, job_name, job_group) references qrtz_job_details(sched_name, job_name, job_group);
alter table qrtz_blob_triggers add primary key (sched_name, trigger_name, trigger_group);
alter table qrtz_blob_triggers add foreign key (sched_name, trigger_name, trigger_group) references qrtz_triggers(sched_name, trigger_name,
trigger_group);
alter table qrtz_cron_triggers add primary key (sched_name, trigger_name, trigger_group);
alter table qrtz_cron_triggers add foreign key (sched_name, trigger_name, trigger_group) references qrtz_triggers(sched_name, trigger_name,
trigger_group);
alter table qrtz_simple_triggers add primary key (sched_name, trigger_name, trigger_group);
alter table qrtz_simple_triggers add foreign key (sched_name, trigger_name, trigger_group) references qrtz_triggers(sched_name,
trigger_name, trigger_group);
alter table qrtz_fired_triggers drop constraint qrtz_fired_triggers_pkey;
alter table qrtz_fired_triggers add primary key (sched_name, entry_id);
```

```
alter table qrtz_calendars drop constraint qrtz_calendars_pkey;
alter table qrtz_calendars add primary key (sched_name, calendar_name);
alter table qrtz_locks drop constraint qrtz_locks_pkey;
alter table qrtz_locks add primary key (sched_name, lock_name);
alter table qrtz_paused_trigger_grps drop constraint qrtz_paused_trigger_grps_pkey;
alter table qrtz_paused_trigger_grps add primary key (sched_name, trigger_group);
alter table qrtz_scheduler_state drop constraint qrtz_scheduler_state_pkey;
alter table qrtz_scheduler_state add primary key (sched_name, instance_name);
-
- add new simprop_triggers table
-
CREATE TABLE qrtz_simprop_triggers
 (
   SCHED_NAME VARCHAR(120) NOT NULL,
   TRIGGER_NAME VARCHAR(200) NOT NULL,
   TRIGGER_GROUP VARCHAR(200) NOT NULL,
   STR_PROP_1 VARCHAR(512) NULL,
   STR_PROP_2 VARCHAR(512) NULL,
   STR_PROP_3 VARCHAR(512) NULL,
   INT_PROP_1 INT NULL,
   INT_PROP_2 INT NULL,
   LONG_PROP_1 BIGINT NULL,
   LONG_PROP_2 BIGINT NULL,
   DEC_PROP_1 NUMERIC(13,4) NULL,
   DEC_PROP_2 NUMERIC(13,4) NULL,
   BOOL_PROP_1 BOOL NULL,
   BOOL_PROP_2 BOOL NULL,
   PRIMARY KEY (SCHED_NAME,TRIGGER_NAME,TRIGGER_GROUP),
   FOREIGN KEY (SCHED_NAME,TRIGGER_NAME,TRIGGER_GROUP)
   REFERENCES QRTZ_TRIGGERS(SCHED_NAME,TRIGGER_NAME,TRIGGER_GROUP)
);
-
- create indexes for faster queries
-
create index idx_qrtz_j_req_recovery on qrtz_job_details(SCHED_NAME,REQUESTS_RECOVERY);
create index idx_qrtz_j_grp on qrtz_job_details(SCHED_NAME,JOB_GROUP);
create index idx_qrtz_t_j on qrtz_triggers(SCHED_NAME,JOB_NAME,JOB_GROUP);
create index idx_qrtz_t_jg on qrtz_triggers(SCHED_NAME,JOB_GROUP);
create index idx_qrtz_t_c on qrtz_triggers(SCHED_NAME,CALENDAR_NAME);
create index idx_qrtz_t_g on qrtz_triggers(SCHED_NAME,TRIGGER_GROUP);
create index idx_qrtz_t_state on qrtz_triggers(SCHED_NAME,TRIGGER_STATE);
create index idx_qrtz_t_n_state on qrtz_triggers(SCHED_NAME,TRIGGER_NAME,TRIGGER_GROUP,TRIGGER_STATE);
create index idx_qrtz_t_n_g_state on qrtz_triggers(SCHED_NAME,TRIGGER_GROUP,TRIGGER_STATE);
create index idx_qrtz_t_next_fire_time on qrtz_triggers(SCHED_NAME,NEXT_FIRE_TIME);
create index idx_qrtz_t_nft_st on qrtz_triggers(SCHED_NAME,TRIGGER_STATE,NEXT_FIRE_TIME);
create index idx_qrtz_t_nft_misfire on qrtz_triggers(SCHED_NAME,MISFIRE_INSTR,NEXT_FIRE_TIME);
create index idx_qrtz_t_nft_st_misfire on qrtz_triggers(SCHED_NAME,MISFIRE_INSTR,NEXT_FIRE_TIME,TRIGGER_STATE);
create index idx_qrtz_t_nft_st_misfire_grp on
qrtz_triggers(SCHED_NAME,MISFIRE_INSTR,NEXT_FIRE_TIME,TRIGGER_GROUP,TRIGGER_STATE);
create index idx_qrtz_ft_trig_inst_name on qrtz_fired_triggers(SCHED_NAME,INSTANCE_NAME);
create index idx_qrtz_ft_inst_job_req_rcvry on qrtz_fired_triggers(SCHED_NAME,INSTANCE_NAME,REQUESTS_RECOVERY);
create index idx_qrtz_ft_j_g on qrtz_fired_triggers(SCHED_NAME,JOB_NAME,JOB_GROUP);
create index idx_qrtz_ft_jg on qrtz_fired_triggers(SCHED_NAME,JOB_GROUP);
create index idx_qrtz_ft_t_g on qrtz_fired_triggers(SCHED_NAME,TRIGGER_NAME,TRIGGER_GROUP);
create index idx_qrtz_ft_tg on qrtz_fired_triggers(SCHED_NAME,TRIGGER_GROUP);
```

2.7 Nouvelles fonctionnalités

Pour l'utilisation des nouvelles fonctionnalités de la version 3.10.2 à savoir :

- Mise à jour de l'utilisation de quartz
- Préconisation d'utilisation des mails
- Listener pour la journalisation d'évènements.
- Intégration de la librairie EHCache
- Ajout de metrologiefilter + Spring AOP

merci de se référer aux chapitres correspondants dans le « Guide du développeur »