

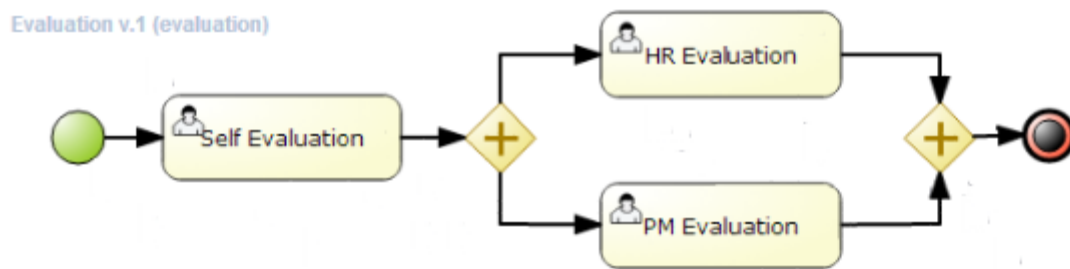


Introduction et démarrage avec jBPM

1. Vue d'ensemble

1.1. Qu'est-ce que jBPM?

jBPM est une suite de gestion de processus métier (BPM) flexible. Il est léger, entièrement open-source (distribué sous Apache License 2.0) et écrit en Java. Il vous permet de modéliser, d'exécuter et de surveiller les processus et les incidents de l'entreprise tout au long de leur cycle de vie.



Un processus métier vous permet de modéliser vos objectifs métier en décrivant les étapes à exécuter pour atteindre ces objectifs. L'ordre de ces objectifs est décrit à l'aide d'un organigramme. Ce processus améliore considérablement la visibilité et l'agilité de votre logique métier. jBPM se concentre sur les processus métier exécutables, qui contiennent suffisamment de détails pour pouvoir être exécutés sur un moteur de BPM. Les processus opérationnels exécutables combleront le fossé entre les utilisateurs professionnels et les développeurs car ils sont de niveau supérieur et utilisent des concepts spécifiques à un domaine qui sont compris par les utilisateurs professionnels mais peuvent également être exécutés directement.

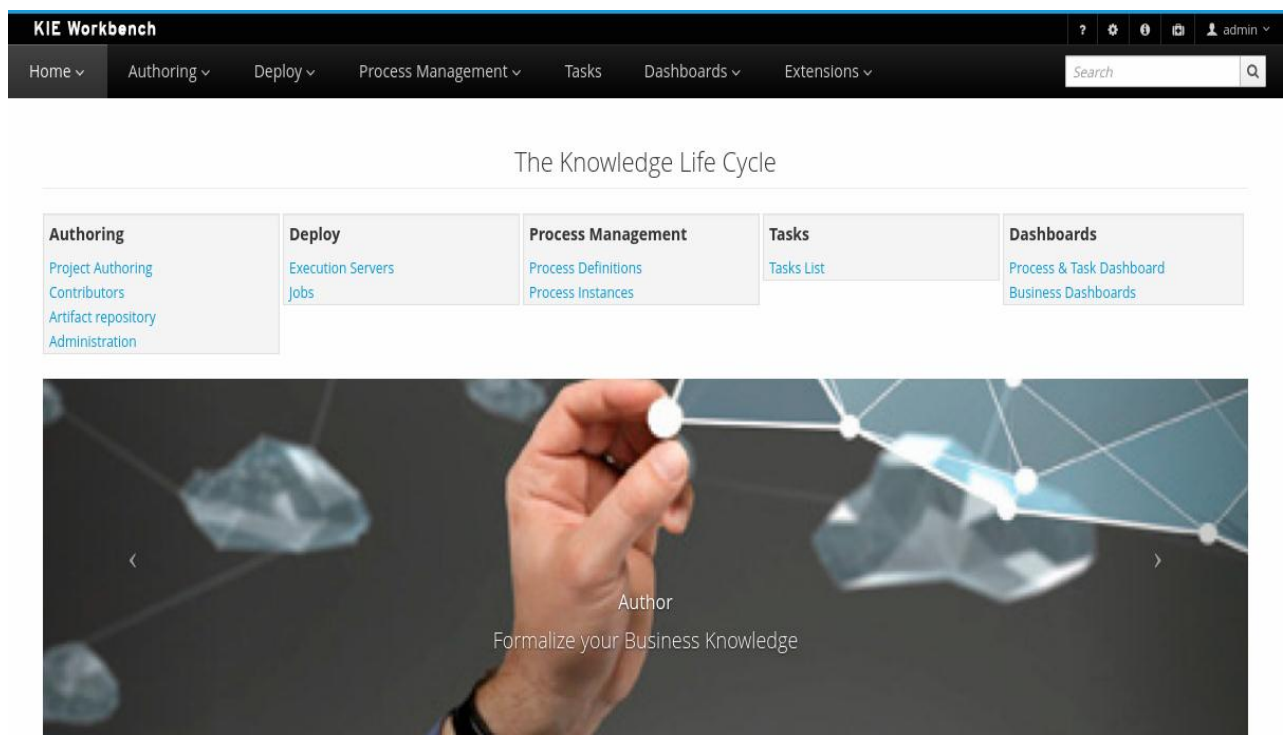
Les processus métier doivent être pris en charge tout au long de leur cycle de vie: création, déploiement, gestion des processus et listes de tâches, tableaux de bord et rapports.

Le noyau de jBPM est un moteur de flux de travail léger et extensible écrit en Java pur qui vous permet d'exécuter des processus métier à l'aide de la dernière [spécification BPMN 2.0](#). Il peut fonctionner dans n'importe quel environnement Java, intégré à votre application ou en tant que service.

Outre le moteur principal, de nombreuses fonctionnalités et outils sont proposés pour prendre en charge les processus métier tout au long de leur cycle de vie:

- Service de tâches utilisateur enfichable basé sur WS-HumanTask pour inclure des tâches devant être exécutées par des acteurs humains.
- Persistance et transactions enfichables (basées sur JPA / JTA).
- Fonctionnalités de gestion de cas ajoutées au moteur principal pour prendre en charge des cas d'utilisation plus adaptatifs et flexibles
- Concepteur de processus basé sur le Web pour prendre en charge la création graphique et la simulation de vos processus d'entreprise (glisser-déposer).

- Modélisateur de données Web et modélisateur de formulaires pour la création de modèles de données et de formulaires de tâches.
- Tableaux de bord et rapports personnalisables basés sur le Web
- Tous combinés dans un atelier basé sur le Web, prenant en charge le cycle de vie complet du MPM:
 - Modélisation et déploiement - créez vos processus, règles, modèles de données, formulaires et autres actifs
 - Exécution: exécute des processus, des tâches, des règles et des événements sur le moteur d'exécution principal.
 - Gestion de l'exécution - travaillez sur la tâche assignée, gérez les instances de processus, etc.
 - Reporting - suivez l'évolution de l'exécution à l'aide des fonctionnalités de Business Activity Monitoring



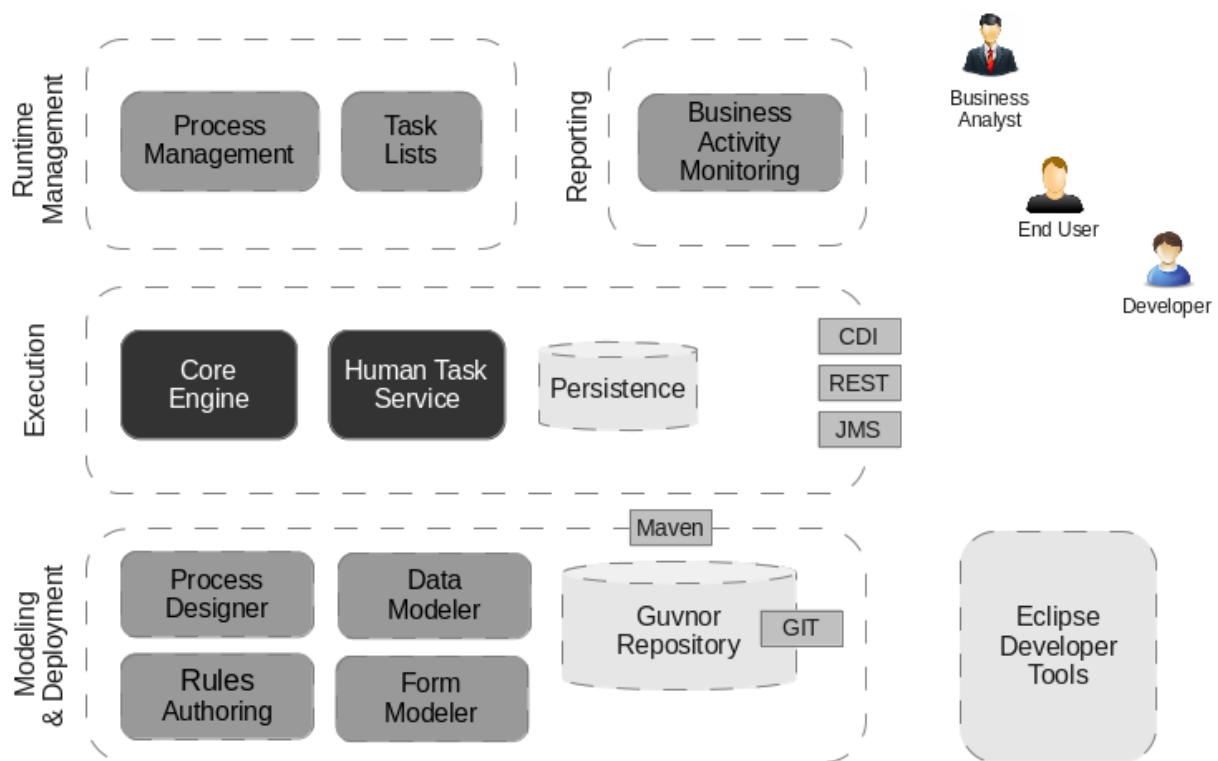
- Outils de développement basés sur Eclipse pour prendre en charge la modélisation, les tests et le débogage des processus
- API distante pour traiter le moteur en tant que service (REST, JMS, API Java distante)
- Intégration avec Maven, Spring, OSGi, etc.

BPM crée un pont entre les analystes, les développeurs et les utilisateurs finaux en proposant des fonctionnalités et des outils de gestion des processus d'une manière qui plaise autant aux utilisateurs qu'aux développeurs. Des noeuds spécifiques à un domaine peuvent être connectés à la palette, ce qui facilite la compréhension des processus par les utilisateurs professionnels.

jBPM prend en charge la gestion de cas en offrant des fonctionnalités plus avancées pour prendre en charge des processus adaptatifs et dynamiques nécessitant de la souplesse pour modéliser des situations réelles et complexes qui ne peuvent pas être facilement décrites à l'aide d'un processus rigide. Nous redonnons le contrôle aux utilisateurs finaux en leur permettant de contrôler les parties du processus à exécuter. Cela permet une déviation dynamique du processus.

jBPM n'est pas simplement un moteur de processus isolé. Une logique métier complexe peut être modélisée comme une combinaison de processus métier avec des règles métier et un traitement d'événements complexe. jBPM peut être associé au [projet Drools](#) pour prendre en charge un environnement unifié intégrant ces paradigmes dans lesquels vous modélisez votre logique métier sous la forme d'une combinaison de processus, de règles et d'événements.

1.2. Vue d'ensemble



Cette figure donne un aperçu des différentes composantes du projet jBPM.

- Le moteur principal est au cœur du projet et vous permet d'exécuter les processus de manière flexible. Il s'agit d'un composant Java pur que vous pouvez choisir d'intégrer à votre application ou de le déployer en tant que service et de vous y connecter via l'interface utilisateur Web ou les API distantes.
 - Un service de base facultatif est le service de tâche humaine qui prend en charge le cycle de vie d'une tâche humaine si des acteurs humains participent au processus.
 - Un autre service de base facultatif est la persistance à l'exécution. Cela conservera l'état de toutes vos instances de processus et enregistrera les informations d'audit relatives à tout ce qui se passe au moment de l'exécution.

- Les applications peuvent se connecter au moteur principal via son API Java ou en tant qu'ensemble de services CDI, mais également à distance via une API REST et JMS.
- Les outils Web vous permettent de modéliser, simuler et déployer vos processus et autres artefacts connexes (tels que des modèles de données, des formulaires, des règles, etc.):
 - Le concepteur de processus permet aux utilisateurs métiers de concevoir et de simuler des processus métier dans un environnement Web.
 - Le modélisateur de données permet aux utilisateurs non techniques d'afficher, de modifier et de créer des modèles de données à utiliser dans vos processus.
 - Un modélisateur de formulaires basé sur le Web vous permet également de créer, générer ou modifier des formulaires liés à vos processus (pour démarrer le processus ou effectuer l'une des tâches de l'utilisateur).
 - La création de règles vous permet de spécifier différents types de règles de gestion (tables de décision, règles guidées, etc.) à combiner avec vos processus.
 - Tous les actifs sont stockés et gérés par le référentiel Guvnor (exposé via Git) et peuvent être gérés (versioning), construits et déployés.
- La console de gestion Web permet aux utilisateurs professionnels de gérer leur environnement d'exécution (gérer les processus d'entreprise, tels que démarrer de nouveaux processus, inspecter les instances en cours, etc.), de gérer leur liste de tâches, d'effectuer des activités et de générer des rapports.
- Les outils de développement basés sur Eclipse sont une extension de l'EDI Eclipse, destinés aux développeurs. Ils vous permettent de créer des processus métier par glisser-déposer, de tester et de déboguer vos processus, etc.

Chacun des composants est décrit plus en détail ci-dessous.

1.3. Moteur de base

Le moteur principal de jBPM est le cœur du projet. C'est un moteur de flux de travail léger qui exécute vos processus métier. Il peut être intégré à votre application ou déployé en tant que service (éventuellement dans le cloud). Ses caractéristiques les plus importantes sont les suivantes:

- Moteur de base solide et stable pour l'exécution de vos instances de processus.
- Prise en charge native de la dernière spécification BPMN 2.0 pour la modélisation et l'exécution de processus métier.
- Focalisation sur la performance et l'évolutivité.
- Léger (peut être déployé sur presque tous les périphériques prenant en charge un environnement Java Runtime simple; aucun conteneur Web n'est requis).
- (Facultatif) persistance connectable avec une implémentation JPA par défaut.
- Prise en charge des transactions enfichables avec une implémentation JTA par défaut.
- Implémenté en tant que moteur de processus générique, il peut donc être étendu pour prendre en charge de nouveaux types de nœuds ou d'autres langages de processus.

- Les auditeurs sont prévenus de divers événements.
- Possibilité de migrer les instances de processus en cours d'exécution vers une nouvelle version de leur définition de processus

Le moteur principal peut également être intégré à quelques autres services principaux (indépendants):

- Le service de tâches utilisateur peut être utilisé pour gérer des tâches humaines lorsque des acteurs humains doivent participer au processus. Il est entièrement enfichable et l'implémentation par défaut est basée sur la spécification WS-HumanTask et gère le cycle de vie des tâches, listes de tâches, formulaires de tâches et certaines fonctionnalités plus avancées telles que l'escalade, la délégation, les affectations basées sur des règles, etc.
- Le journal d'historique peut stocker toutes les informations sur l'exécution de tous les processus du moteur. Cela est nécessaire si vous avez besoin d'accéder aux informations historiques, car la persistance à l'exécution ne stocke que l'état actuel de toutes les instances de processus actives. Le journal d'historique peut être utilisé pour stocker tous les états actuels et historiques d'instances de processus actives et terminées. Il peut être utilisé pour rechercher des informations relatives à l'exécution d'instances de processus, à des fins de surveillance, d'analyse, etc.

1.4. Process Designer

Le concepteur jBPM basé sur le Web vous permet de modéliser vos processus d'entreprise dans un environnement Web. Il est destiné aux utilisateurs professionnels et offre un éditeur graphique permettant d'afficher et de modifier vos processus d'entreprise (par glisser-déposer), similaire au plug-in Eclipse. Il prend en charge les allers-retours entre l'éditeur Eclipse et le concepteur basé sur le Web. Il prend également en charge la simulation de processus.

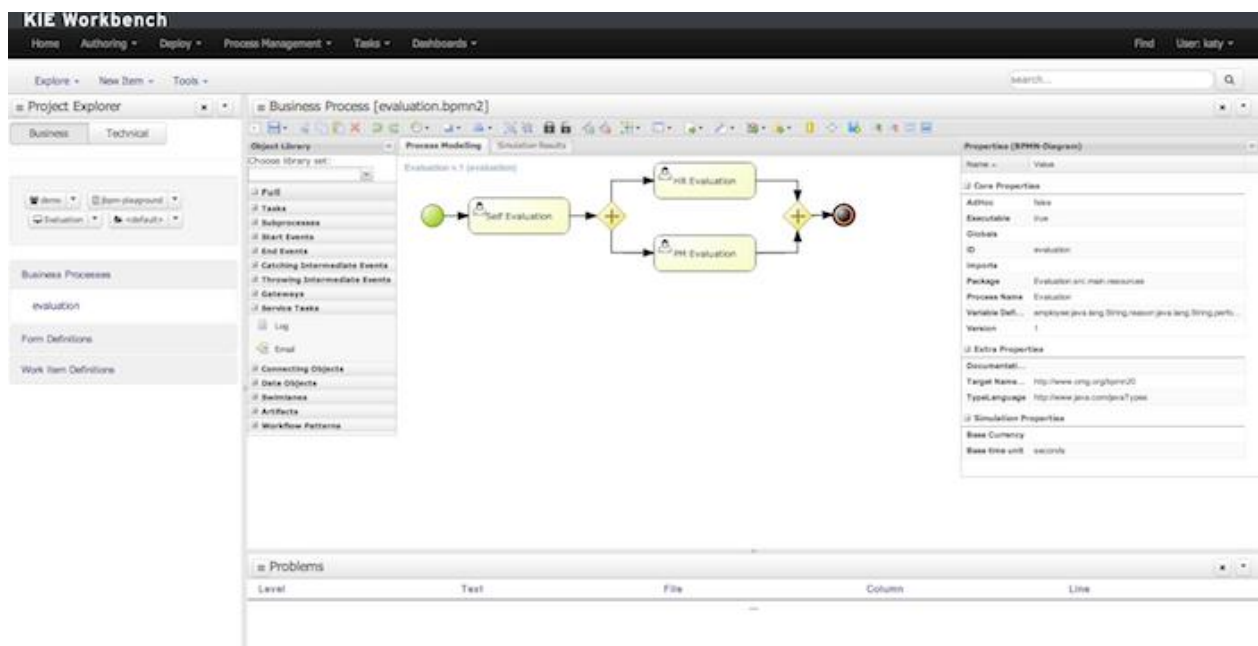


Figure 1. Concepteur Web pour la création de processus BPMN2

1.5 Modélisateur de données

Les processus ont presque toujours des données avec lesquelles travailler. Le modélisateur de données permet aux utilisateurs non techniques d'afficher, de modifier ou de créer ces modèles de données.

En règle générale, un analyste de processus métier ou un analyste de données saisit les exigences d'un processus ou d'une application et les transforme en un ensemble formel de structures de données interdépendantes. Le nouvel outil Data Modeler fournit une aide simple, directe et visuelle pour la création de modèles de données logiques et physiques, sans nécessiter de compétences de développement avancées ni de codage explicite. Le modélisateur de données est intégré de manière transparente dans le workbench. Ses principaux objectifs sont de faire des modèles de données des citoyens de première classe un processus d'amélioration du processus et de permettre une automatisation complète du processus grâce à l'utilisation intégrée des structures de données (et des formulaires qui seront utilisés pour interagir avec celles-ci).

1.6. Formulaire de modèle

JBPM Form Modeler est un moteur de formulaire et un éditeur qui permet aux utilisateurs de créer des formulaires pour capturer et afficher des informations lors de l'exécution d'un processus ou d'une tâche, sans avoir besoin de compétences en codage ou en marquage de modèle.

Il fournit un environnement WYSIWYG pour modéliser des formulaires faciles à utiliser pour les utilisateurs moins expérimentés.

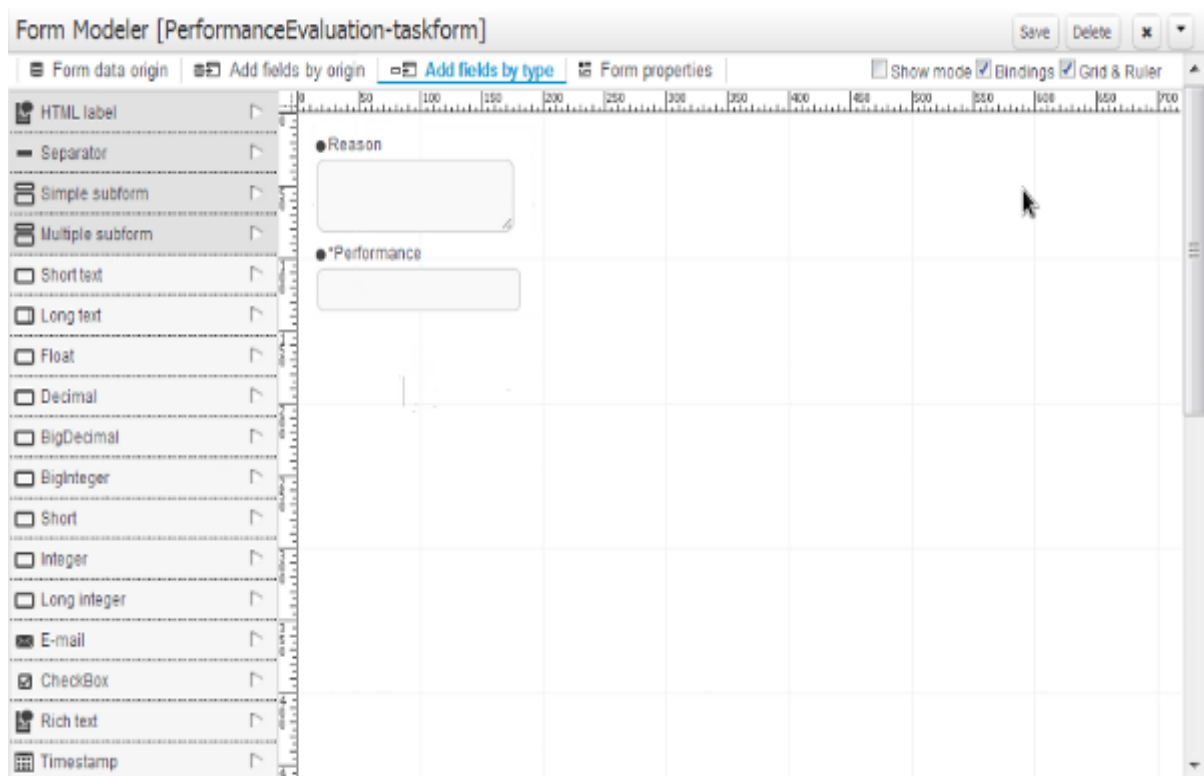


Figure 2. Formulaire de modélisation

Principales caractéristiques:

- Modélisation de formulaire WYSIWYG UI pour les formulaires
- Génération automatique de formulaire à partir d'objets de modèle de données / Java
- Liaison de données pour les objets Java
- Formule et expressions
- Formats de formulaires personnalisés
- Formes incorporant

Les interfaces utilisateur du modélisateur de formulaire visent à la fois les analystes de processus et les développeurs pour la création et le test de formulaires.

Les développeurs ou les utilisateurs avancés peuvent également utiliser des fonctionnalités avancées pour personnaliser le comportement et l'apparence des formulaires.

1.7. Instance de processus et gestion des tâches

Les processus métier peuvent être gérés via une console de gestion Web. Il est destiné aux utilisateurs professionnels et ses principales fonctionnalités sont les suivantes:

- Gestion d'instances de processus: possibilité de démarrer de nouvelles instances de processus, d'obtenir une liste des instances de processus en cours d'exécution, d'inspecter visuellement l'état d'une instance de processus spécifique.
- Gestion des tâches humaines: être en mesure d'obtenir une liste de toutes vos tâches en cours (qu'elles vous ont été attribuées ou que vous pourrez éventuellement réclamer) et d'exécuter les tâches de votre liste de tâches (à l'aide de formulaires de tâches personnalisables).

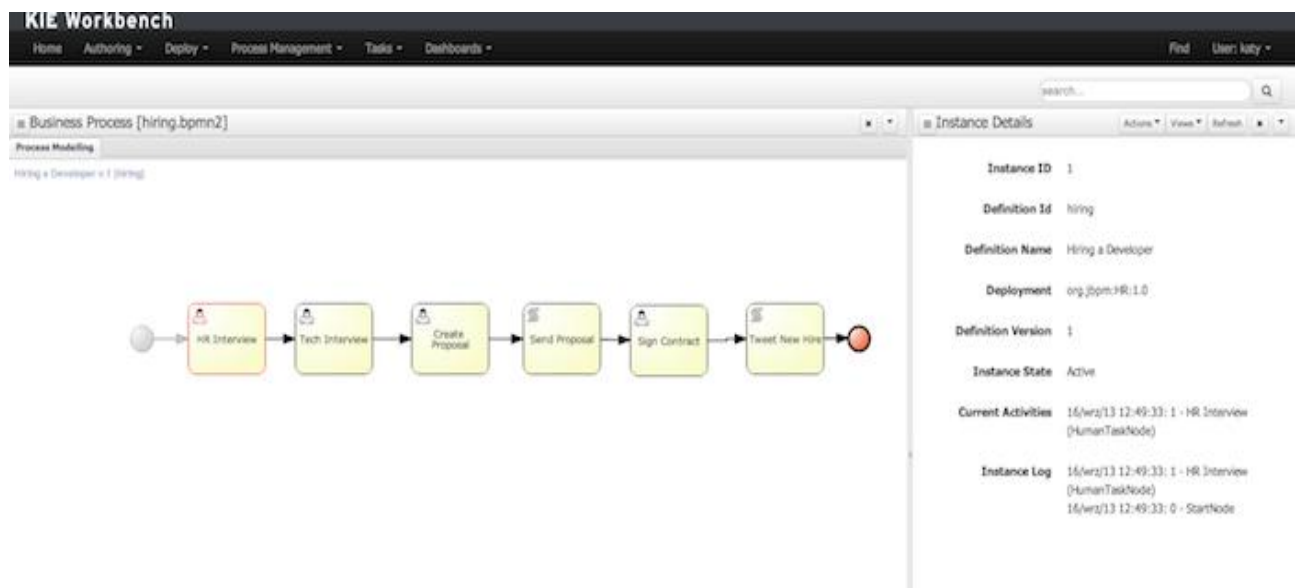


Figure 3. Gestion de vos instances de processus

1.8. Surveillance de l'activité commerciale

À compter de la version 6.0, jBPM est fourni avec un outil BAM complet qui permet aux utilisateurs non techniques de composer visuellement des tableaux de bord professionnels. Avec ce tout nouveau module, développer des solutions de surveillance de l'activité commerciale et de création de rapports sur jBPM n'a jamais été aussi simple!

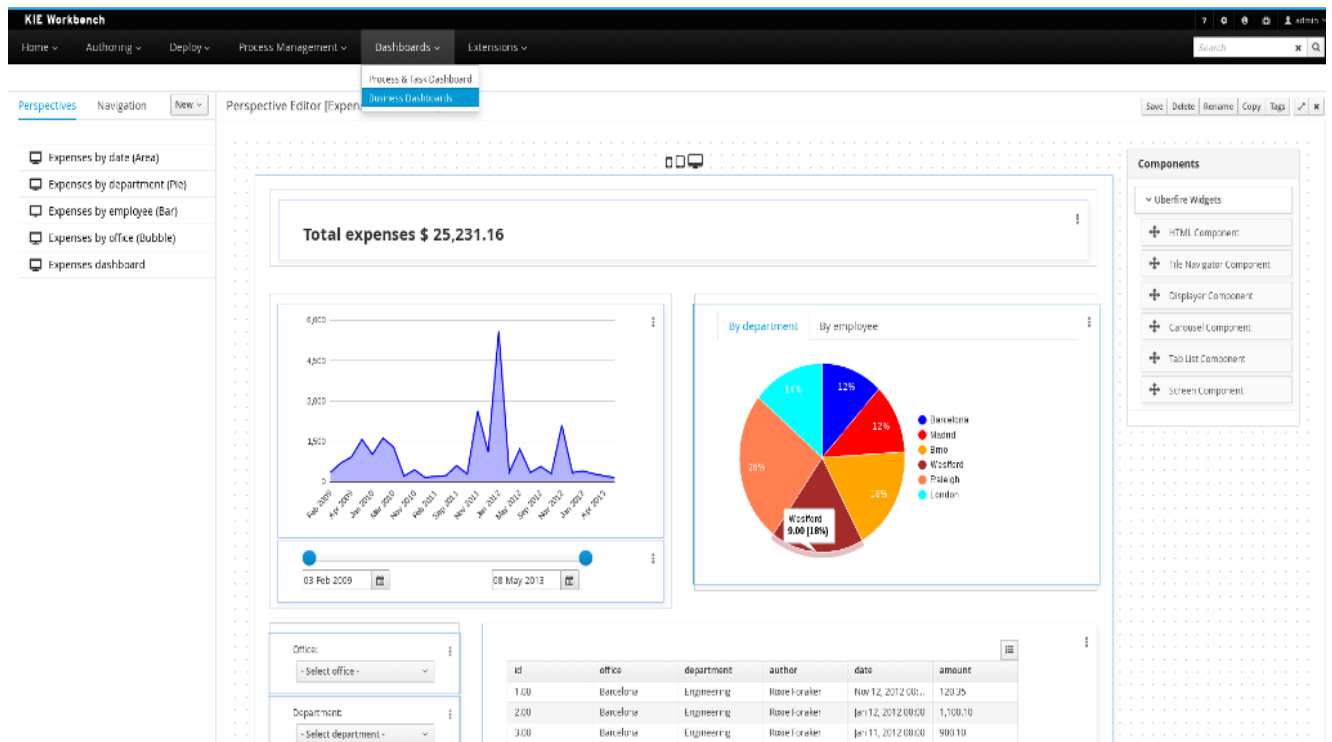


Figure 4. Surveillance de l'activité commerciale

Principales caractéristiques:

- Configuration visuelle des tableaux de bord (Drag'n'drop).
- Représentation graphique des indicateurs de performance clés (KPI).
- Configuration des tables de rapport interactives.
- Exportation de données au format Excel et CSV.
- Filtrage et recherche, en mémoire ou en SQL.
- Extraction de données à partir de systèmes externes, via différents protocoles.
- Contrôle d'accès granulaire pour différents profils d'utilisateurs.
- Look'n'feel des outils de personnalisation.
- Architecture de bibliothèque de cartes enfichable.

Utilisateurs cibles:

- Gestionnaires / propriétaires d'entreprise. Consommateur de tableaux de bord et de rapports.

- Architectes informatiques / systèmes. Connectivité et extraction de données.
- Analystes / Développeurs. Composition et configuration du tableau de bord.

Pour plus d'informations sur les nouvelles et remarquables fonctionnalités BAM de jBPM, veuillez consulter le chapitre Surveillance des activités commerciales .

1.9 Table de travail

Le workbench est une application Web qui combine tous les outils Web ci-dessus dans une solution configurable.

Il prend en charge les éléments suivants:

- Un service de référentiel pour stocker vos processus métier et les artefacts associés, à l'aide d'un référentiel Git, qui prend en charge la gestion des versions, l'accès Git distant (en tant que système de fichiers) et l'accès via REST.
- Une interface utilisateur basée sur le Web pour gérer vos processus d'affaires, ciblée sur les utilisateurs professionnels; il prend également en charge la visualisation (et l'édition) de vos artefacts (les éditeurs basés sur le Web tels que Designer, les modélisateurs de données et de formulaires sont intégrés ici), mais également la catégorisation, la construction et le déploiement, etc.
- Fonctionnalités de collaboration permettant à plusieurs acteurs (utilisateurs et développeurs, par exemple) de travailler ensemble sur le même projet.

L'application Workbench couvre le cycle de vie complet des projets de BPM à partir de la phase de création, en passant par la mise en œuvre, l'exécution et le suivi.

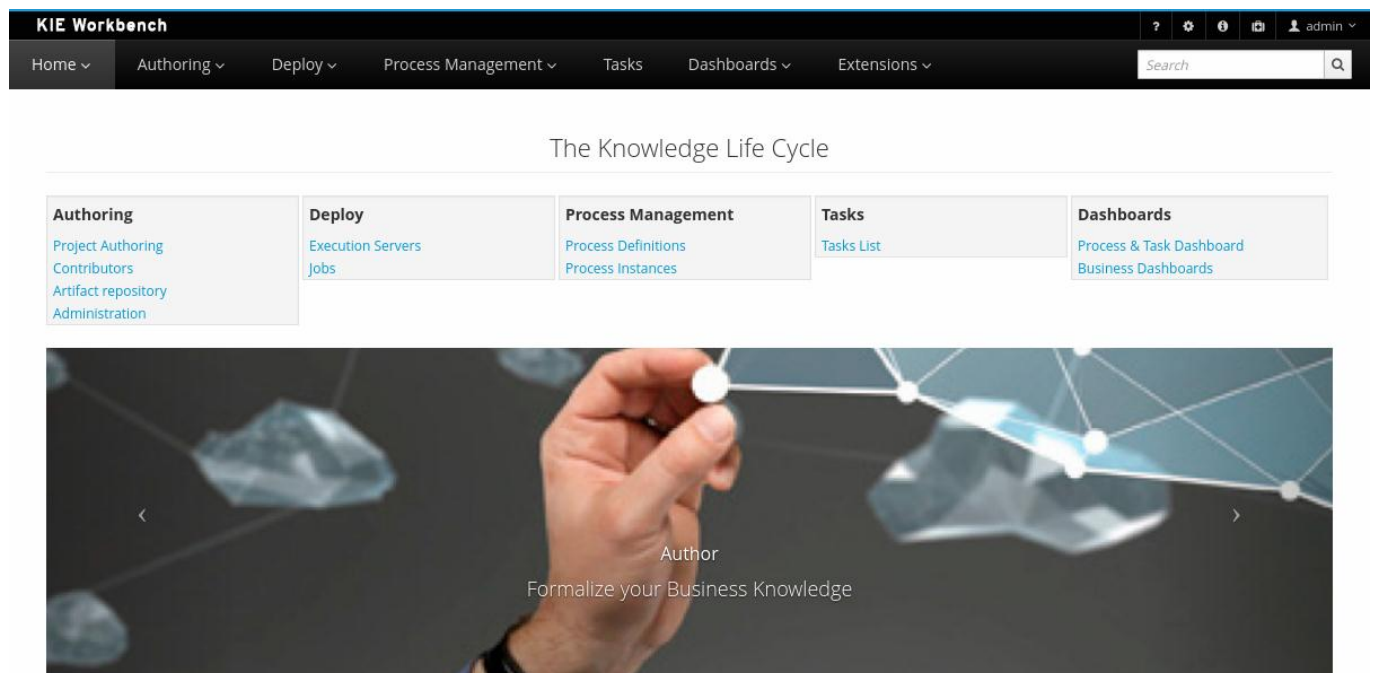


Figure 5. Application KIE Workbench

1.10. Outils de développement Eclipse

Les outils basés sur Eclipse constituent un ensemble de plug-ins à l'IDE Eclipse et vous permettent d'intégrer vos processus métier dans votre environnement de développement. Il est destiné aux développeurs et propose des assistants, un éditeur graphique pour la création de vos processus d'entreprise (par glisser-déposer) et de nombreuses fonctionnalités de test et de débogage avancées.

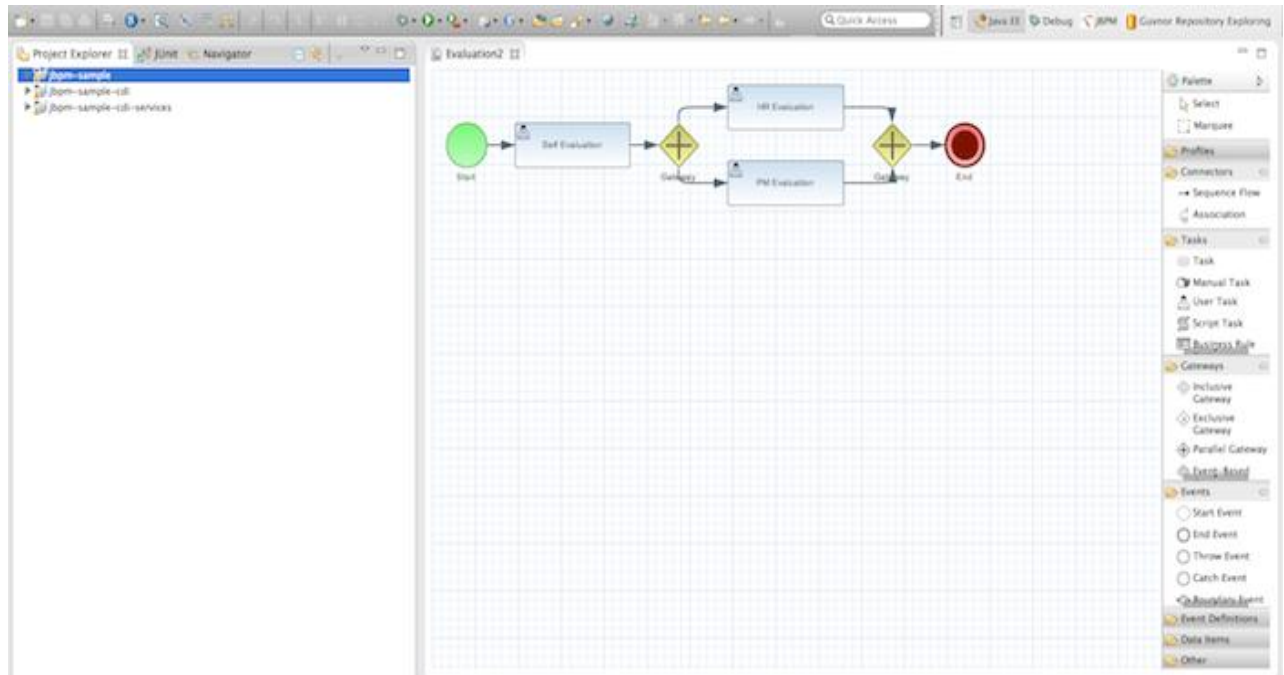


Figure 6. Editeur Eclipse pour la création de processus BPMN2

Il comprend les fonctionnalités suivantes:

- Assistant pour créer un nouveau projet jBPM
- Un éditeur graphique pour les processus BPMN 2.0
- La possibilité de connecter vos propres nœuds spécifiques à un domaine
- Validation
- Support Runtime (vous pouvez donc sélectionner la version de jBPM que vous souhaitez utiliser)
- Débogage graphique pour voir toutes les instances de processus en cours d'une session sélectionnée, pour visualiser l'état actuel d'une instance de processus spécifique, etc.

2. Commencer

Si vous souhaitez suivre un didacticiel rapide qui vous guidera à travers la plupart des composants à l'aide d'un exemple simple, consultez le chapitre Installateur . Cela vous apprendra à télécharger et à utiliser le programme d'installation pour créer une configuration de démonstration comprenant la plupart des composants. Il utilise un exemple simple pour vous

guider à travers les fonctionnalités les plus importantes. Des screencasts sont également disponibles pour vous aider.

Si vous souhaitez lire d'abord plus d'informations, les chapitres suivants se concentrent d'abord sur le moteur principal (API, BPMN 2.0, etc.). Les chapitres suivants décrivent ensuite les autres composants et d'autres sujets plus complexes tels que les processus spécifiques à un domaine, les processus flexibles, etc. Après avoir lu les chapitres principaux, vous devriez être en mesure de passer directement à d'autres chapitres qui pourraient vous intéresser.

Vous pouvez également commencer à jouer avec quelques exemples proposés dans un téléchargement séparé. Consultez le chapitre Exemples pour voir comment commencer à jouer avec ceux-ci.

Après avoir lu ces chapitres, vous devriez être prêt à créer vos propres processus et à intégrer le moteur à votre application. Ces processus peuvent être lancés à partir du programme d'installation ou à partir de zéro.

2.1. Téléchargements

Les dernières versions peuvent être téléchargées à partir de jbpm.org . Il suffit de choisir l'artefact que vous voulez:

- **bin** : tous les fichiers binaires jBPM (JAR) et leurs dépendances transitives
- **src** : les sources des composants de base
- **docs** : la documentation
- **exemples** : quelques exemples jBPM, peuvent être importés dans Eclipse
- **programme d'installation** : le programme d'installation de jBPM télécharge et installe une configuration de démonstration de jBPM
- **installer-complet** : le programme d'installation complet de jBPM, télécharge et installe une version de démonstration de jBPM, contient déjà un certain nombre de dépendances préemballées (elles ne doivent donc pas être téléchargées séparément)

2.2. Communauté

Voici de nombreux liens utiles faisant partie de la communauté jBPM:

- Le compte Twitter [#jbossjbpm](https://twitter.com/jbossjbpm) .
- Configuration de jBPM et forums d'utilisateurs et listes de diffusion sur l'utilisation de jBPM
- Un système de suivi des bogues JIRA pour les bogues, les demandes de fonctionnalités et la feuille de route

N'hésitez pas à nous rejoindre sur notre canal IRC à l'adresse [#jbpm](http://chat.freenode.net). C'est là que se déroulent la plupart des discussions en temps réel sur le projet et que vous pouvez également trouver la plupart des développeurs. Vous n'avez pas de client IRC installé? Rendez-vous simplement sur <http://webchat.freenode.net/> , entrez le surnom souhaité et spécifiez [#jbpm](http://webchat.freenode.net/#jbpm). Cliquez ensuite sur connexion pour vous joindre à la fête.

2.3. Sources

2.3.1. Licence

Le code jBPM lui-même utilise la licence Apache v2.0.

Certains autres composants que nous intégrons ont leur propre licence:

- Le nouveau plug-in Eclipse BPMN2 est Eclipse Public License (EPL) v1.0.
- Le concepteur Web est basé sur Oryx / Wapama et possède une licence MIT.
- Le projet Drools est Apache License v2.0.

2.3.3. Construire à partir de la source

Si vous êtes intéressé par la construction du code source, la contribution, la publication, etc., assurez-vous de lire ce fichier README .

2.4. Être impliqué

On nous demande souvent "Comment puis-je participer". Heureusement, la réponse est simple: il suffit d'écrire du code et de le soumettre :) Il n'y a pas d'obstacles à franchir ni de poignées de main secrètes. Nous demandons une "surcharge" très minime pour permettre un développement de projet évolutif. Vous trouverez ci-dessous un aperçu général des outils et du "flux de travail" que nous demandons, ainsi que des conseils généraux.

Si vous faites du bon travail, n'oubliez pas de bloguer à ce sujet :)

2.4.1. Inscrivez-vous sur jboss.org

En vous inscrivant sur jboss.org, vous aurez accès au wiki, aux forums et à JIRA de JBoss. Allez sur <https://www.jboss.org/> et cliquez sur "S'inscrire".



2.4.2. Signer l'accord de contributeur

Le seul formulaire à signer est l'accord de contribution, entièrement automatisé via le Web. Comme le montre l'image ci-dessous "Cela établit les conditions générales de vos contributions et garantit que le code source peut être concédé sous licence de manière appropriée"

Sign CLA

If you've submitted a patch that's been accepted, or been offered an invitation to commit directly into a project's source code repository, then please login using your jboss.org user account and sign an [Individual](#) or [Corporate](#) Contributor License Agreement (CLA).

This establishes the terms and conditions for your contributions and ensures that the source code can be licensed appropriately.

Username:

Password:

Login



Do not sign a CLA unless you've met the conditions above.

This helps to keep our systems tidy and prevents project leads from reviewing unnecessary agreements.

2.4.3. Soumission de problèmes via JIRA

Pour pouvoir interagir avec l'équipe de développement principale, vous devez utiliser JIRA, l'outil de suivi des problèmes. Cela garantit que toutes les demandes sont consignées et attribuées à un calendrier de publication et à toutes les discussions capturées au même endroit. Les rapports de bogues, les corrections de bogues, les demandes de fonctionnalités et les soumissions de fonctionnalités doivent tous être placés ici. Les questions générales doivent être posées sur les listes de diffusion.

Les envois de code mineurs, tels que les corrections de format ou de documentation, n'ont pas besoin de créer un problème JIRA associé.

<https://issues.jboss.org/browse/DROOLS>

<https://issues.jboss.org/browse/JBPM>

<https://issues.jboss.org/browse/GUVNOR>

Dashboards ▾ Projects ▾ Issues ▾ Agile ▾


Drools / JBRULES-3370

Array fields are not supported in declared facts

Log In

Details

Type:  Enhancement

Priority:  Minor

Affects Version/s: None

Component/s: drools-compiler, drools-core

Labels: None

Status:  Open (View Workflow)

Resolution: Unresolved

Fix Version/s: None

Security Level: **Public** (Everyone can see)

Similar Issues: [Show 10 results >](#)

Description

it should be possible to do

```

declare Bean
arrayField : SomeObject[]
end


optionally,

declare Bean
arrayField : SomeObject[] = new SomeObject[3]
end

```

2.4.4. Fork GitHub

Avec l'accord de contributeur signé et vos demandes soumises à JIRA, vous devriez maintenant être prêt à coder :) Créez un compte GitHub et branchez l'un des dépôts Drools, jBPM ou Guvnor. La fork en créera une copie dans votre propre espace GitHub sur lequel vous pourrez travailler à votre rythme. Si vous faites une erreur, ne vous inquiétez pas, faites-la disparaître et fourchez à nouveau. Notez que chaque référentiel GitHub vous fournit l'URL de clone (paiement), GitHub vous fournira des URL spécifiques à votre fork.


droolsjbpm / drools

Admin Watch Fork Pull Request 125 81

Code
Network
Pull Requests 10
Stats & Graphs

Drools Expert is the rule engine and Drools Fusion does complex event processing (CEP). — [Read more](#)
<http://www.jboss.org/drools>

ZIP SSH HTTP Git Read-Only

git@github.com:droolsjbpm/drools.git

Read+Write access

branch: master ▾
Files
Commits
Branches 4
Tags 10
Downloads

2.4.5. Tests d'écriture

Lorsque vous écrivez des tests, essayez de les garder minimales et autonomes. Nous préférons conserver les fragments DRL dans le test, car cela permet une révision plus rapide. S'il existe un grand nombre de règles, l'utilisation d'une chaîne n'est pas pratique. Dans ce cas, placez-les dans des fichiers DRL distincts au lieu d'être chargées à partir du chemin d'accès aux classes. Si vos tests doivent utiliser un modèle, essayez d'utiliser ceux qui existent déjà pour d'autres tests

unitaires, tels que personne, fromage ou ordre. S'il n'existe aucune classe contenant les champs dont vous avez besoin, essayez de mettre à jour les champs des classes existantes avant d'ajouter une nouvelle classe.

Il y a un grand nombre de tests à examiner pour se faire une idée, MiscTest est un bon point de départ.

```


557 @Test
558 public void testEvalWithBigDecimal() throws Exception {
559     String str = "";
560     str += "package org.drools \n";
561     str += "import java.math.BigDecimal; \n";
562     str += "global java.util.List list \n";
563     str += "rule rule1 \n";
564     str += "    dialect \"java\" \n";
565     str += "when \n";
566     str += "    $bd : BigDecimal() \n";
567     str += "    eval( $bd.compareTo( BigDecimal.ZERO ) > 0 ) \n";
568     str += "then \n";
569     str += "    list.add( $bd ); \n";
570     str += "end \n";
571
572     KnowledgeBuilder kbuilder = KnowledgeBuilderFactory.newKnowledgeBuilder();
573
574     kbuilder.add( ResourceFactory.newByteArrayResource( str.getBytes() ),
575                 ResourceType.DRL );
576
577     if ( kbuilder.hasErrors() ) {
578         logger.warn( kbuilder.getErrors().toString() );
579     }
580     assertFalse( kbuilder.hasErrors() );
581
582     KnowledgeBase kbase = KnowledgeBaseFactory.newKnowledgeBase();
583     kbase.addKnowledgePackages( kbuilder.getKnowledgePackages() );
584
585     StatefulKnowledgeSession ksession = createKnowledgeSession(kbase);
586     List list = new ArrayList();
587     ksession.setGlobal( "list",
588                       list );
589     ksession.insert( new BigDecimal( 1.5 ) );
590
591     ksession.fireAllRules();
592
593     assertEquals( 1,
594                  list.size() );
595     assertEquals( new BigDecimal( 1.5 ),
596                  list.get( 0 ) );
597 }
598

```


2.4.6. S'engager avec les conventions correctes

Lorsque vous vous engagez, veuillez à utiliser les conventions appropriées. Le commit doit commencer par l'identifiant du problème JIRA, tel que JBRULES-220. Cela garantit que les validations sont référencées via JIRA afin que nous puissions voir toutes les validations pour un problème donné au même endroit. Après l'identifiant, le titre du problème devrait venir ensuite. Utilisez ensuite une nouvelle ligne, mise en retrait avec un tiret, pour fournir des informations supplémentaires relatives à ce commit. Utilisez une nouvelle ligne et un tiret supplémentaires pour chaque point distinct que vous souhaitez faire valoir. Vous pouvez ajouter des références croisées JIRA supplémentaires au même commit, si cela convient. En général, essayez d'éviter de combiner des problèmes non liés dans le même commit.

N'oubliez pas de modifier votre base de données locale à partir du maître d'origine, puis repoussez vos commits vers votre base de données.

 Drools / JBRULES-328 FactTemplates / JBRULES-329

implement core handling of Templates for ObjectType

Log In

▼ mark.proctor@jboss.com submitted changeset 5421 to trunk in JBossRules (29 files) - 02/Aug/06 8:14 PM

[JBRULES-220](#) Refactor ObjectType to work with Templates
-This also involved refactor Evaluator to use Enums for ValueType and Operator

[JBRULES-329](#) implement core handling of Templates for ObjectType
-Initial commit for FactTemplate work, still not integrated into parsers and builds, it also needs unit tests.

[JBRULES-246](#) Allow & and | connectives for field constraints
-XmlReader is now fixed
-Xml and Drl Dumpers have been fixed

- trunk/drools-compiler/src/main/java/org/drools/lang/DrlDumper.java (+53 -27) ▲ □ 🔍 ↕
- trunk/drools-compiler/src/main/java/org/drools/lang/descr/FieldConstraintDescr.java (+5 -1) ▲ □ 🔍 ↕
- trunk/drools-compiler/src/main/java/org/drools/lang/descr/LiteralRestrictionDescr.java (+7 -7) ▲ □ 🔍 ↕
- trunk/drools-compiler/src/main/java/org/drools/lang/descr/ReturnValueRestrictionDescr.java (+7 -9) ▲ □ 🔍 ↕
- trunk/drools-compiler/src/main/java/org/drools/semantics/java/RuleBuilder.java (+74 -62) ▲ □ 🔍 ↕
- trunk/drools-compiler/src/main/java/org/drools/xml/BoundVariableHandler.java (+0 -110) ▲ □ 🔍 ↕
- trunk/drools-compiler/src/main/java/org/drools/xml/FieldBindingHandler.java (+2 -6) ▲ □ 🔍 ↕
- trunk/drools-compiler/src/main/java/org/drools/xml/FieldConstraintHandler.java (+95) ▲ □ 🔍 ↕
- trunk/drools-compiler/src/main/java/org/drools/xml/LiteralHandler.java (+0 -110) ▲ □ 🔍 ↕
- trunk/drools-compiler/src/main/java/org/drools/xml/LiteralRestrictionHandler.java (+103) ▲ □ 🔍 ↕

...19 more files in changeset

▼ Mark Proctor <mdproctor@gmail.com> submitted changeset b98d43508c91f1cb01d53b22395693ca87d69d5e to 5.2.x in 8:14 PM

[JBRULES-220](#) Refactor ObjectType to work with Templates -This also involved refactor Evaluator to use Enums for Value
[JBRULES-329](#) implement core handling of Templates for ObjectType
-Initial commit for FactTemplate work, still not integrated into parsers and builds, it also needs unit tests.

[JBRULES-246](#) Allow & and | connectives for field constraints
-XmlReader is now fixed
-Xml and Drl Dumpers have been fixed

2.4.7. Soumettre des demandes de tirage

Avec votre code rebasé à partir du maître d'origine et poussé vers votre zone GitHub personnelle, vous pouvez maintenant soumettre votre travail sous forme de demande d'extraction. Si vous regardez le haut de la page dans GitHub pour votre zone de travail, il s'agira d'un bouton "Demande d'extraction". En sélectionnant cette option, vous obtiendrez une interface utilisateur pour automatiser la soumission de votre demande d'extraction.

La demande d'extraction est ensuite placée dans une file d'attente que tout le monde peut voir et commenter. Ci-dessous, vous pouvez voir une demande de tir typique. Les demandes d'extraction permettent les discussions et affichent tous les validations associées et les différences pour chaque validation. Les discussions impliquent généralement des révisions de code qui fournissent des suggestions utiles pour des améliorations et nous permettent de laisser des commentaires en ligne sur des parties spécifiques du code. Ne soyez pas découragé si nous ne fusionnons pas tout de suite, il peut souvent falloir plusieurs révisions avant d'accepter une demande d'extraction. Heureusement, avec GitHub, il est très facile de revenir à votre code, de faire d'autres commits, puis de mettre à jour votre demande d'extraction à la dernière et à la perfection.

Nous pouvons avoir besoin de temps pour répondre aux demandes reçues, alors soyez patient. Les tests soumis accompagnant un correctif seront généralement appliqués assez rapidement, alors que de simples tests seront souvent utilisés jusqu'à ce que nous ayons le temps de le soumettre également avec un correctif. N'oubliez pas de redéfinir votre requête et de la soumettre à nouveau de temps en temps, sinon, il y aura des conflits de fusion et les principaux développeurs les ignoreront.

The screenshot shows a GitHub pull request interface. At the top, a green bar indicates the pull request is 'Open' and shows the merge path: 'sotty wants someone to merge 5 commits into droolsjbpm:master from sotty:master'. Below this, tabs for 'Discussion', 'Commits', and 'Diff' are visible. The main content area shows the pull request title 'JBRULES-3370 Array fields are not supported in declared facts' and a description: 'Well, not exactly a ground-breaking feature, but still useful :) Also improves bean initialization with MVEL expression'. On the right, a green 'Open' button is shown along with statistics: '+ 388 additions' and '- 60 deletions'. Below the description, a comment from 'etirelli' is visible, dated '22 days ago', stating: '@sotty thanks for providing this. I was reviewing the code, and with a few changes it can also support multi-dimensional arrays (e.g. Object[], int[], etc). Do you think you can change it for that?'. Below the comment, a diff view is shown for the file 'drools-compiler/src/main/java/org/drools/lang/DRLParser.java', with line numbers 924 to 926. A 'View full changes' link is present. At the bottom, another comment from 'etirelli' is shown, dated '22 days ago', stating: 'There is already a rule called type(). Please use that instead of creating a fieldType() rule. It supports multi-dimensional arrays and generics, although I know MVEL does not support generics yet.'.

2.5. Que faire si je rencontre des problèmes ou si j'ai des questions?

Vous pouvez toujours contacter la communauté jBPM pour obtenir de l'aide.

IRC: #jbpm à chat.freenode.net

jBPM Setup Google Group - Discussions sur l'installation, la configuration, la configuration et l'administration des environnements Workbench, Eclipse, d'exécution et des architectures d'entreprise générales.

Utilisation de jBPM Groupe Google - Création, exécution et gestion de processus avec jBPM. Toutes les questions concernant l'utilisation de jBPM. Aide générale sur les API et meilleures pratiques pour la création de systèmes BPM.

Legacy jBPM User Forum - sert d'archive; envoyer de nouvelles questions à l'un des groupes Google ci-dessus

3. jBPM Installer

3.1. Conditions préalables

Ce script suppose que vous avez Java JDK 1.8+ (défini comme JAVA_HOME) et Ant 1.9+ installé. Sinon, utilisez les liens suivants pour les télécharger et les installer:

Java: <http://java.sun.com/javase/downloads/index.jsp>

Ant: <http://ant.apache.org/bindownload.cgi>

Pour vérifier si Java et Ant sont correctement installés, tapez les commandes suivantes dans une invite de commande:

```
java -version
```

```
fourmi -version
```

Cela devrait renvoyer des informations sur la version de Java et Ant que vous utilisez actuellement.

3.2. Télécharger l'installateur

Tout d'abord, vous devez [télécharger](#) le programme d'installation et le décompresser sur votre système de fichiers local. Il y a deux versions

- programme d'installation complet - contient déjà de nombreuses dépendances nécessaires lors de l'installation

- programme d'installation minimal - contient uniquement le programme d'installation et télécharge toutes les dépendances requises à la volée

En général, il est probablement préférable de télécharger le programme d'installation complet: jBPM-7.1.0.Final-installer-full.zip

Vous pouvez également télécharger la [dernière version](#) (uniquement pour l'installateur minimal).

3.3. Configuration de la démonstration

Le moyen le plus simple de commencer consiste simplement à exécuter le script d'installation pour installer la configuration de démonstration. L'installation de démonstration installera tous les outils Web (au-dessus de WildFly) et les outils Eclipse dans une configuration préconfigurée. Accédez au dossier jbpmm-installer dans lequel vous avez décompressé le programme d'installation et exécuté (à partir d'une invite de commande):

```
ant install.demo
```

Cette volonté:

- Télécharger le serveur d'applications WildFly
- Configurer et déployer un serveur d'exécution de processus
- Configurer et déployer le workbench
- Configurer et déployer l'application de gestion de cas
- Télécharger Eclipse
- Installer le plugin Drools et jBPM Eclipse
- Installer le modélisateur Eclipse BPMN 2.0

L'exécution de cette commande peut prendre un certain temps (VRAIMENT, sans blague, nous téléchargeons par exemple une installation Eclipse, même si vous avez téléchargé le programme d'installation complet, spécifiquement pour votre système d'exploitation).

Le script indique toujours le fichier en cours de téléchargement (vous pouvez par exemple vérifier s'il est toujours en cours de téléchargement en vérifiant si la taille du fichier en question dans le dossier jbpmm-installer / lib augmente toujours). Si vous souhaitez éviter de télécharger des composants spécifiques (parce que vous ne les utiliserez pas ou que vous les avez déjà installés ailleurs), vérifiez ci-dessous si vous ne souhaitez exécuter que des parties spécifiques de la démonstration ou diriger le programme d'installation vers un composant déjà installé.

Une fois la configuration de la démo terminée, vous pouvez commencer à jouer avec les différents composants en lançant la configuration de la démo:

```
ant start.demo
```

Cette volonté:

- Démarrer le serveur de base de données H2
- Démarrer le serveur d'applications WildFly
- Démarrer Eclipse

Maintenant, attendez que la console de gestion de processus s'affiche:

<http://localhost:8080/jbpm-console>

L'interface utilisateur de gestion de cas sera disponible sur:

<http://localhost:8080/jbpm-casemgmt>

Le démarrage du serveur d'applications et de l'application Web peut prendre une minute. Si la page Web ne s'affiche pas au bout d'un moment, assurez-vous qu'aucun pare-feu ne bloque ce port, ou une autre application utilisant déjà le port 8080. Vous pouvez toujours consulter le journal du serveur {jbpm-installer- dossier} / wildfly- {version} /standalone/log/server.log

Une fois que tout est démarré, vous pouvez commencer à jouer avec Eclipse et les outils Web, comme expliqué dans les sections suivantes.

Si vous souhaitez uniquement essayer les outils Web et ne souhaitez pas télécharger et installer les outils Eclipse, vous pouvez utiliser ces commandes alternatives:

```
ant install.demo.noecclipse  
ant start.demo.noecclipse
```

De même, si vous souhaitez uniquement essayer les outils Eclipse et ne souhaitez pas télécharger et installer les outils Web, vous pouvez utiliser ces commandes alternatives:

```
ant install.demo.eclipse  
ant start.demo.eclipse
```

Continuez maintenant avec les tutoriels de 10 minutes. Une fois que vous avez fini de jouer et que vous souhaitez arrêter la configuration de la démonstration, vous pouvez utiliser:

```
ant stop.demo
```

Si, à un moment quelconque, vous souhaitez recommencer avec une configuration de démonstration claire, ce qui signifie que tous les changements apportés dans l'outil Web et / ou enregistrés dans la base de données seront perdus, vous pouvez exécuter la commande suivante (après quoi, vous pouvez exécuter le installer à nouveau à partir de zéro, notez que cela ne peut pas être annulé):

```
ant clean.demo
```

3.4. Tutoriel de 10 minutes utilisant le Workbench

Ouvrez la console de gestion de processus:

<http://localhost:8080/jbpm-console>

Le démarrage du serveur d'applications et de l'application Web peut prendre une minute. Si la page Web ne s'affiche pas au bout d'un moment, assurez-vous qu'aucun pare-feu ne bloque ce port, ou une autre application utilisant déjà le port 8080. Vous pouvez toujours consulter le journal du serveur {jbpm-installer- dossier} / wildfly- {version} /standalone/log/server.log

Connectez-vous en utilisant krisv / krisv comme nom d'utilisateur / mot de passe.

À l'aide d'un exemple d'évaluation prédéfini, le [screencast suivant](#) donne un aperçu de la gestion de vos instances de processus. Cela vous montre:

- Comment se connecter au workbench
- Comment importer un exemple de projet existant, le construire et le déployer
- Comment démarrer une nouvelle instance de processus
- Comment rechercher le statut actuel d'une instance de processus en cours d'exécution
- Comment rechercher vos tâches
- Comment compléter une tâche
- Comment consulter les rapports pour surveiller l'exécution de votre processus

Le workbench prend en charge l'ensemble du cycle de vie de vos processus métier: création, déploiement, gestion des processus, tâches et tableaux de bord.

- La perspective de création de projet vous permet d'examiner les référentiels existants, où chaque projet peut contenir des processus métier (mais également des règles métier, des modèles de données, des formulaires, etc.). Il vous permet de créer votre propre projet ou d'importer un exemple existant à examiner.
 - Dans ce screencast, nous commençons par importer le projet Evaluation
- L'explorateur de projet affiche tous les artefacts disponibles:
 - évaluation: processus d'entreprise décrivant le processus d'évaluation comme une séquence de tâches
 - evaluation-taskform: formulaire de processus pour démarrer le processus d'évaluation
 - PerformanceEvaluation-taskform: formulaire de tâche pour effectuer les tâches d'évaluation
- Pour rendre un processus disponible pour l'exécution, vous devez d'abord le créer et le déployer avec succès. Pour ce faire, ouvrez le projet sélectionné (dans la perspective Création du projet) et cliquez sur Construire et déployer (coin supérieur droit).

- Pour gérer vos définitions de processus et vos instances, cliquez sur l'option de menu "Gestion des processus" dans la barre de menus supérieure et sélectionnez l'une des options disponibles en fonction de vos intérêts:
 - Définitions de processus - répertorie toutes les définitions de processus disponibles
 - Instances de processus - répertorie toutes les instances de processus actives (permet d'afficher les processus terminés et abandonnés en modifiant les critères de filtrage)
- La vue Définitions de processus vous permet de démarrer une nouvelle instance de processus en cliquant sur le bouton Démarrer. Le formulaire de processus (tel que défini dans le projet) s'affiche. Vous devez y renseigner les informations nécessaires pour lancer le processus. Dans ce cas, vous devez renseigner l'utilisateur pour lequel vous souhaitez démarrer une évaluation (par exemple, utilisez "krisv") et indiquez le motif de la demande, après quoi vous pourrez remplir le formulaire. Certains détails sur l'instance de processus qui vient d'être démarrée seront affichés dans le panneau de détails d'instance de processus. De là, vous pouvez accéder à des détails supplémentaires:
 - Modèle de processus - pour visualiser l'état actuel du processus
 - Variables de processus - pour voir les valeurs actuelles des variables de processus
 - Documents - documents liés à l'instance de processus
 - Journaux - aperçu de tous les événements de processus pour cette instance

L'instance de processus que vous venez de démarrer nécessite tout d'abord une auto-évaluation de l'utilisateur et attend que l'utilisateur ait terminé cette tâche.

- Pour voir les tâches qui vous ont été assignées, choisissez l'option de menu "Tâches" sur la barre du haut. Par défaut, toutes les tâches actives seront affichées et une "évaluation de performance" (créée par l'instance de processus que vous venez de démarrer) devrait vous être proposée. Lorsque vous cliquez sur une tâche, les détails de la tâche s'affichent, y compris le formulaire de tâche associé à cette tâche. Après avoir démarré la tâche, vous pouvez renseigner les informations nécessaires et terminer la tâche. Une fois la tâche terminée, vous pouvez vérifier une nouvelle fois les "instances de processus" pour vérifier la progression de votre instance de processus. Vous devriez être en mesure de constater que le processus attend maintenant que votre responsable des ressources humaines et votre responsable de projet procèdent également à une évaluation. Vous pouvez vous connecter en tant que "john" / "john" et "mary" / "mary" pour effectuer ces tâches.
- Après avoir démarré et / ou complété quelques instances de processus et tâches humaines, vous pouvez générer un rapport sur ce qui s'est passé jusqu'à présent. Sous "Tableaux de bord", sélectionnez "Tableau de bord de processus et de tâches". Il s'agit d'un ensemble de graphiques prédéfinis qui permettent aux utilisateurs de voir ce qui se passe dans le système. Les graphiques peuvent également être entièrement personnalisés, comme expliqué dans le chapitre Surveillance des activités commerciales.

3.5 Tutoriel de 10 minutes avec Eclipse

Le [screencast suivant](#) donne un aperçu de l'utilisation des outils Eclipse. Cela vous montre:

- Comment importer et exécuter l'exemple de projet d'évaluation
 - Importer le projet d'évaluation (inclus dans jbpmm-installer)

- Ouvrez le processus Evaluation.bpmn
- Ouvrez la classe Java com.sample.ProcessTest.
- Exécuter la classe ProcessTest pour exécuter le processus
- Comment créer un nouveau projet jBPM (avec exemple de processus et test JUnit)

Vous pouvez importer le projet d'évaluation - un exemple inclus dans jbpmm-installer - en sélectionnant "Fichier → Importer ...", sélectionnez "Projets existants dans l'espace de travail" et accédez au dossier jbpmm-installer / sample / evaluation puis cliquez sur "Terminer". . Vous pouvez ouvrir le processus d'évaluation et la classe ProcessTest. Pour exécuter la classe, cliquez dessus avec le bouton droit de la souris et sélectionnez "Exécuter en tant que... - Application Java". La console doit montrer comment le processus a été lancé et comment les différents acteurs du processus ont exécuté les tâches qui leur ont été affectées, afin de compléter l'instance de processus.

Vous pouvez également créer un nouveau projet à l'aide de l'assistant de projet jBPM. Les exemples de projets contiennent un processus et un fichier Java associé pour démarrer le processus. Sélectionnez "Fichier - Nouveau... - Projet ..." et sous la catégorie "jBPM", puis sélectionnez "Projet jBPM". Sélectionnez cette option pour créer un projet avec quelques exemples de fichiers afin de vous permettre de démarrer rapidement, puis cliquez sur Suivant. Donnez un nom au projet. Vous pouvez choisir parmi un exemple simple HelloWorld ou un exemple légèrement plus avancé utilisant la persistance et les tâches humaines. Si vous sélectionnez ce dernier et cliquez sur Terminer, vous devriez voir un nouveau projet contenant un processus "sample.bpmn" et une classe de test "com.sample.ProcessTest". Vous pouvez ouvrir le processus BPMN2 en double-cliquant dessus. Pour exécuter le processus, cliquez avec le bouton droit sur ProcessTest.java et sélectionnez "Exécuter en tant que - application Java".

3.6. Configuration

3.6.1. Authentification de l'atelier

L'application Web du plan de travail utilise l'*autre* domaine de sécurité pré-installé pour authentifier et autoriser les utilisateurs (comme indiqué dans le *fichier WEB-INF / jboss-web.xml* à l'intérieur des fichiers WAR).

Le serveur d'applications utilise par défaut des domaines basés sur des fichiers de propriétés - Veuillez noter que cette configuration est uniquement destinée à des fins de démonstration (les utilisateurs, les rôles et les mots de passe sont stockés dans de simples fichiers de propriétés sur le système de fichiers).

L'authentification est configurée dans le fichier *standalone.xml* comme suit:

```
<security-domain name="other" cache-type="default">
  <authentication>
    <login-module code="Remoting" flag="optional">
      <module-option name="password-stacking"
value="useFirstPass"/>
    </login-module>
```



```

        <login-module code="RealmDirect" flag="required">
            <module-option name="password-stacking"
value="useFirstPass"/>
        </login-module>
        <login-module
code="org.kie.security.jaas.KieLoginModule" flag="optional"
module="deployment.jbpm-console.war"/>
    </authentication>
</security-domain>
<security-realm name="ApplicationRealm">
    <authentication>
        <local default-user="$local" allowed-users="*" skip-
group-loading="true"/>
        <properties path="users.properties" relative-
to="jboss.server.config.dir"/>
    </authentication>
    <authorization>
        <properties path="roles.properties" relative-
to="jboss.server.config.dir"/>
    </authorization>
</security-realm>

```

Ce sont les utilisateurs par défaut:

Tableau 1. Utilisateurs par défaut

prénom	Mot de passe	Rôles de l'atelier	Rôles de la tâche
admin	admin	administrateur, analyste, kiemgmt, reste-tout, kie-server	
Krisv	Krisv	administrateur, analyste, reste-tout, kie-server	
John	John	analyste, kie-server	Comptabilité, PM
Marie	Marie	analyste, kie-server	HEURE
représentant commercial	représentant commercial	analyste, kie-server	Ventes
jack	jack	analyste, kie-server	IL
Katy	Katy	analyste, kie-server	HEURE
Salaboy	Salaboy	administrateur, analyste, reste-tout, kie-server	TI, RH, comptabilité

prénom	Mot de passe	Rôles de l'atelier	Rôles de la tâche
Kieserver	kieserver1!	kie-server	

L'authentification peut être personnalisée à l'aide de l'une des options suivantes:

- Écrans de gestion des utilisateurs et des groupes sur l'application Web du plan de travail.

Naviguez dans l'application Web du plan de travail, cliquez sur le menu *Accueil* → *Admin* et sélectionnez *Utilisateurs* .

- Le script *add-user* fourni par défaut sur Wildfly / EAP.

Exemple pour les plates-formes Linux - exécutez la commande suivante et suivez les instructions du script:

```
/bin/sh $JBOSS_HOME/bin/add-user.sh
--user-properties
$JBOSS_HOME/standalone/configuration/users.properties
--group-properties
$JBOSS_HOME/standalone/configuration/roles.properties
--realm ApplicationRealm
```

3.6.2. Utilisation de votre propre base de données avec le programme d'installation de jBPM

3.6.2.1. introduction

jBPM utilise la spécification de l'API Java Persistence (v2) pour permettre aux utilisateurs de configurer la source de données qu'ils souhaitent utiliser pour conserver les données d'exécution. Par conséquent, les instructions ci-dessous décrivent comment configurer une source de données lors de l'utilisation de JPA sur un serveur d'applications JBoss (par exemple, EAP7 ou Wildfly10) à l'aide d'un fichier *persistence.xml* et de la configuration de votre source de données et de votre pilote dans le fichier *standalone.xml* de votre serveur d' *applications* . vous configureriez toute autre application utilisant JPA sur le serveur d'applications. Le programme d'installation automatise certaines de ces tâches (comme copier les bons fichiers au bon endroit après l'installation).

Par défaut, jbpn-installer utilise une base de données H2 pour la persistance des données d'exécution. Dans cette section nous allons:

1. modifier les paramètres de persistance pour la persistance d'exécution de l'état d'instance de processus
2. Testez le démarrage avec nos nouveaux paramètres!

Vous aurez besoin d'une instance locale d'une base de données. Dans ce cas, nous utiliserons MySQL.

3.6.2.2. Configuration de la base de données

Dans la base de données MySQL utilisée dans ce quickstart, créez un utilisateur unique:

- utilisateur / schéma "jbpm" avec le mot de passe "jbpm" (qui sera utilisé pour conserver toutes les entités)

Si vous vous retrouvez avec des noms différents pour vos utilisateurs / schémas, veuillez noter où nous insérons "jbpm" dans les fichiers de configuration.

Si vous souhaitez essayer ce démarrage rapide avec *une autre* base de données, une section à la fin de ce démarrage rapide décrit ce que vous devrez peut-être modifier.

3.6.2.3. Configuration

Les fichiers suivants définissent les paramètres de persistance pour la démo de jbpmm-installer:

- jbpmm-installer / db / jbpmm-persistence-JPA2.xml
- Configuration du serveur d'application
 - autonome - *. xml

Plusieurs fichiers standalone.xml sont disponibles (selon que vous utilisez JBoss EAP ou Wildfly et que vous exécutiez le profil normal ou complet). Le profil complet est requis pour utiliser le composant JMS pour l'intégration à distance. Il sera donc utilisé par défaut par le programme d'installation. La meilleure pratique consiste à mettre à jour tous les fichiers standalone.xml afin de garantir une configuration cohérente, mais le plus important est de disposer de standalone-full-wildfly- {version} .xml correctement configuré, car il est utilisé par défaut par le programme d'installation.

Faites ce qui suit:

- Désactiver la base de données H2 par défaut et activer la base de données MySQL dans build.properties
- # default is H2
- # H2.version=1.3.168
- # db.name=h2
- # db.driver.jar.name=\${db.name}.jar
- #
db.driver.download.url=http://repo1.maven.org/maven2/com/h2database/h2/\${H2.version}/h2-\${H2.version}.jar
- #mysql
- db.name=mysql
- db.driver.module.prefix=com/mysql
- db.driver.jar.name=mysql-connector-java-5.1.18.jar
db.driver.download.url=https://repository.jboss.org/nexus/service/local/repositories/central/content/mysql/mysql-connector-java/5.1.18/mysql-connector-java-5.1.18.jar

Vous voudrez peut-être mettre à jour le nom du fichier jar du pilote de base de données et télécharger l'URL vers la version du fichier jar qui correspond à votre installation.

- *db/jbpm-persistence-JPA2.xml* :

Il s'agit du fichier de persistance JPA qui définit les paramètres de persistance utilisés par jBPM pour les informations de moteur de processus, les informations de journalisation / BAM et le service de tâches.

Dans ce fichier, vous devrez changer le nom du dialecte de veille prolongée utilisé pour votre base de données.

La ligne d'origine est:

```
<property name="hibernate.dialect"
value="org.hibernate.dialect.H2Dialect"/>
```

Dans le cas d'une base de données MySQL, vous devez la changer en:

```
<property name="hibernate.dialect"
value="org.hibernate.dialect.MySQLDialect"/>
```

Pour ceux d'entre vous qui ont décidé d'utiliser une autre base de données, une liste des classes de dialecte hibernate disponibles peut être trouvée [ici](#) .

- *standalone-full-wildfly- {version} .xml* :

Standalone.xml et *standalone-full.xml* constituent la configuration du serveur d'applications JBoss autonome. Lorsque le programme d'installation installe la démonstration, il copie ces fichiers dans le *standalone/configuration* répertoire situé dans le répertoire du serveur JBoss. Étant donné que le programme d'installation utilise Wildfly par défaut en tant que serveur d'applications, vous devez probablement modifier *standalone-full-wildfly- {version} .xml* .

Nous devons modifier la configuration de la source de données *standalone-full.xml* afin que le moteur de processus jBPM puisse utiliser notre base de données MySQL. Le fichier original contient (quelque chose de très similaire à) les lignes suivantes:

```
<datasource jta="true" jndi-
name="java:jboss/datasources/jbpmDS" pool-name="H2DS"
enabled="true" use-java-context="true" use-ccm="true">
  <connection-url>jdbc:h2:tcp://localhost/~jbpm-
db;MVCC=TRUE</connection-url>
  <driver>h2</driver>
  <security>
    <user-name>sa</user-name>
  </security>
</datasource>
<drivers>
  <driver name="h2" module="com.h2database.h2">
    <xa-datasource-
class>org.h2.jdbcx.JdbcDataSource</xa-datasource-class>
  </driver>
</drivers>
```

Modifiez les lignes comme suit:

```
<datasource jta="true" jndi-
name="java:jboss/datasources/jbpmDS" pool-name="MySQLDS"
enabled="true" use-java-context="true" use-ccm="true">
  <connection-
url>jdbc:mysql://localhost:3306/jbpm</connection-url>
  <driver>mysql</driver>
  <security>
    <user-name>jbpm</user-name>
    <password>jbpm</password>
  </security>
</datasource>
```

et ajoutez une configuration de pilote supplémentaire:

```
<driver name="mysql" module="com.mysql">
  <xa-datasource-
class>com.mysql.jdbc.jdbc2.optional.MysqlXADataSource</xa-
datasource-class>
</driver>
```

- Pour installer les fichiers de pilotes dans le serveur d'applications JBoss (Wildfly, EAP, etc.), il est recommandé d'installer le fichier de pilotes en tant que module. Le programme d'installation s'occupe déjà de la plupart des *tâches* : il copie le *fichier jar* du pilote (spécifié dans le *build.properties*) dans le bon dossier du répertoire modules de votre serveur et y place un module.xml correspondant. Pour MySQL, ce fichier s'appelle *db / mysql_module.xml* . Ouvrez ce fichier et assurez-vous que le nom du fichier jar du pilote répertorié est identique au nom du fichier jar du pilote que vous avez spécifié dans le *fichier build.properties* (y compris la version). Notez que même si vous avez simplement supprimé la configuration par défaut de MySQL, vous devrez toujours ajouter la bonne version ici.

- **Commencer la démo**

Nous avons modifié tous les fichiers nécessaires à ce stade. Ce serait le bon moment pour vous assurer que votre base de données est également démarrée!

Le script d'installation copie ce fichier dans le fichier WAR de jbpm-console avant que celui-ci ne soit installé sur le serveur. Si vous avez déjà exécuté le programme d'installation, il est recommandé de l'arrêter et de le nettoyer en utilisant

```
ant stop.demo
et
```

```
ant clean.demo
avant de continuer.
```

Courir

```
ant install.demo
```

pour (ré) installer les guerres et copier les fichiers de configuration nécessaires. Une fois que vous avez fait cela, démarrez la démo en utilisant

```
ant start.demo
```

- **Problèmes?**

Si cela ne fonctionne pas pour vous, essayez ce qui suit:

- Veuillez vérifier les fichiers que vous avez modifiés: j'ai *écrit* ceci, mais j'ai quand même fait des erreurs lors du changement de fichiers!

- Assurez-vous de ne pas exécuter secrètement une autre instance (non modifiée) de JBoss AS.
- Si aucune de ces méthodes ne fonctionne (et que vous utilisez MySQL), veuillez nous en informer.

3.6.2.4. Utiliser une base de données différente

Si vous décidez d'utiliser une base de données différente avec cette démo, vous devez vous rappeler les points suivants lorsque vous suivez les étapes ci-dessus:

- Configuration de la source de données jBPM dans `standalone.xml`:
 - Après avoir localisé la `java:jboss/datasources/jbpmDS` source de données, vous devez fournir les propriétés suivantes spécifiques à votre base de données:
 - Changer l'URL de votre base de données
 - Changer le nom d'utilisateur et le mot de passe
 - Changer le nom du pilote (que vous allez créer ensuite)

Par exemple:

```
<datasource jta="true" jndi-
name="java:jboss/datasources/jbpmDS" pool-
name="PostgreSQLDS" enabled="true" use-java-
context="true" use-ccm="true">
  <connection-
url>jdbc:postgresql://localhost:5432/jbpm</connection-
url>
  <driver>postgresql</driver>
  <security>
    <user-name>jbpm</user-name>
    <password>jbpm</password>
  </security>
</datasource>
```

- Ajoutez une configuration de pilote supplémentaire:
 - Modifiez le nom du pilote pour qu'il corresponde au nom que vous avez spécifié lors de la configuration de la source de données à l'étape précédente.
 - Modifiez le module du pilote: le fichier de pilote de base de données doit être installé en tant que module (voir ci-dessous) et vous devez ici faire référence au nom unique du module. Étant donné que l'installateur peut s'occuper de générer automatiquement ce module pour vous (voir ci-dessous), il doit correspondre à la `db.driver.module.prefix` propriété dans `build.properties` (où les barres obliques sont remplacées par un point). Dans l'exemple ci-dessous, j'ai utilisé `org/postgresqlcomme` `db.driver.module.prefix` ce qui signifie que je devrais ensuite utiliser `org.postgresqlcomme` nom de module pour le pilote.
 - Entrez le nom correct de la classe de source de données XA à utiliser.

Par exemple:

+

```
<driver name="postgresql" module="org.postgresql">
  <xa-datasource-
class>org.postgresql.xa.PGXADatasource</xa-datasource-class>
</driver>
```

- Vous devez remplacer le dialecte dans *persistence.xml* par celui de votre base de données, par exemple:

```
<property name="hibernate.dialect"
value="org.hibernate.dialect.PostgreSQLDialect"/>
```

- Afin de vous assurer que votre pilote sera correctement installé sur le serveur d'applications JBoss, il existe généralement plusieurs options, telles que l'installation en tant que module ou en tant que déploiement. Il est recommandé d'installer le pilote en tant que module pour EAP et Wildfly.
 - [Installez](#) le fichier JAR du pilote en tant que *module* , comme le fait le script d'installation.
 - [Sinon, vous pouvez modifier et installer](#) le fichier JAR téléchargé en tant que *déploiement* . Dans ce cas, vous devrez copier le fichier JAR vous-même dans le standalone/deployments répertoire.

Si vous choisissez d'installer le pilote en tant que module JBoss (recommandé), procédez comme suit:

- Dans `build.properties`, désactivez les propriétés du pilote H2 par défaut.
 - # default is H2
 - # H2.version=1.3.168
 - # db.name=h2
 - # db.driver.jar.name=h2-\${H2.version}.jar
 - # db.driver.download.url=http://repo1.maven.org/maven2/com/h2database/h2/\${H2.version}/h2-\${H2.version}.jar
- Décommentez l'un des autres exemples de configuration (mysql ou postgresql) ou créez le vôtre:
 - #postgresql
 - db.name=postgresql
 - db.driver.module.prefix=org/postgresql
 - db.driver.jar.name=postgresql-9.1-902.jdbc4.jar
 - db.driver.download.url=https://repository.jboss.org/nexus/content/repositories/thirdparty-uploads/postgresql/postgresql/9.1-902.jdbc4/postgresql-9.1-902.jdbc4.jar
- Remplacez la `db.name` propriété par `build.properties` un nom pour votre base de données.
- Attribuez à la `db.driver.module.prefix` propriété un nom pour le module de votre pilote. Notez que cela doit correspondre à la propriété du module lors de la configuration du pilote dans *standalone.xml* (les barres obliques dans le préfixe ici sont remplacées par un point). Dans l'exemple ci-dessus, j'ai utilisé `org/postgresql` comme `db.driver.module.prefix` ce qui signifie

que je devrais ensuite utiliser `org.postgresql` comme nom de module pour le pilote.

- Remplacez la `db.driver.jar.name` propriété par le nom du fichier jar contenant votre pilote de base de données.
- Modifiez la `db.driver.download.url` propriété à l'emplacement où le fichier JAR du pilote peut être téléchargé. Vous pouvez également télécharger le fichier manuellement vous-même et le placer dans le `db/drivers` dossier, en utilisant le même nom que celui spécifié dans la `db.driver.jar.name` propriété.
- Enfin, vous devrez créer le `db/${db.name}_module.xml` fichier. Par exemple, vous pouvez utiliser `db/mysql_module.xml`, faites-en une copie et:
 - Changez le nom du *module* pour qu'il corresponde au nom du *module* de pilote ci-dessus.
 - Remplacez le nom du chemin de ressource du module par le nom de la `db.driver.jar.name` propriété.
- Par exemple, le haut du fichier ressemblerait à ceci:

```
<module xmlns = "urn:jboss:module:1.0" name =  
"org.postgresql">  
  <resources>  
    <resource-root path = "postgresql-9.1-902.jdbc4.jar" />  
  </resources>
```