Instalacao Server v103

Tags: indi server site

Table of Contents [-]

Introdução

O Intellector-NG é uma solução para a gestão de processos decisórios, fornecendo uma interação fácil através de uma linguagem simples e prática. Este manual descreve passa-a-passo o processo de instalaç

Para quem se destina o manual

O público alvo para este manual são administradores de sistemas e *application servers*, e que tenha bom conhecimento de JBoss e JVM.

O que é necessário para entender esse manual

Este manual assume que os administradores tem familiaridade com os conceitos de administração e *deployment* do JBoss, Linux ou Windows e Sun JDK.

Dicas, advertências, etc.

Nós iremos fazer uso dos seguintes pictogramas:

Informações como **'Dica'** podem ser úteis por várias razões: economia de tempo, diminuição de riscos, etc.

Você deverá ter cuidado em informações marcadas como **Importante**. Ignorando tais informações, definitivamente não é uma boa ideia.

advertências não deveriam ser ignoradas...

Pré-requisitos

Como um primeiro passo, verifique e tenha certeza de qual versão do **Intellector-NG** você deseja instalar ou atualizar. Baseado na versão, você pode determinar a compatibilidade apropriada das versões de J

Se for você imprimir este manual, aproveite e escreva abaixo as versões da sua instalação:

- Intellector-NG: v----------
- Linux, Windows, etc: v----------
- JBoss: v----------
- Java JDK: v----------

JDK 6.0

Instale a Sun JDK 6.0 apropriada para sua plataforma. As JDKs estão disponíveis no seguinte site:http://java.sun.com/javase/downloads/index.jsp

A versão atual do Intellector-NG **SOMENTE** funciona com a **Java SE Development Kit (JDK) 6 Update 10** ou superior, recomendamos enfaticamente, usar superior ao Update 10. O Intellector-NG NÃO FUNC

Não é suficiente ter somente o *Java Runtime Environment (JRE)*; você realmente precisa do **full-blown JDK**

Não se esqueça de configurar **JAVA_HOME** no seu ambiente, mas, forneceremos um *bootstrap* pro Intellector-NG, onde essa variável é configurada, mas você pode querer colocá-la em outro lugar; não se esq

- Embora a localização da instalação da JDK seja livre, sugerimos criar uma pasta *'<drive>:\opt'* no Windows ou usar o *'/opt'* no Unix; ajuda para propósitos de suporte.

Instalando em sistemas Unix, tenha bastante cuidado com as permissões de **escrita** para *owner/group* para o diretório do JBoss, pois o Intellector-NG vai precisar **'escrever'** nesse diretório.

JBoss

O Intellector-NG foi testado nas versões JBoss 4.2.2-GA e 4.2.3-GA; não foi testado ainda na versão 5RC atual. Você pode fazer o download do JBoss, para qualquer plataforma do seguinte site:http://www.jbo

Este é um bom lugar para tirar dúvidas e procurar dicas sobre o JBoss...http://wiki.jboss.org/

Instalando...

Após instalar a Java JDK e o JBoss nos locais apropriados, faça um pequeno teste de verificação da instalação, subindo o JBoss (JBOSS_HOME/bin/run.sh ou run.bat no Windows); para verificar se tudo foi be *application server*.

Evite instalar JDK e JBoss em diretórios que contenham espaços no nome, como "C:\Arquivo de Programas" do Windows.

conheça mais sobre JBoss Enterprise Application Platform.

Bootstrap para JBoss/Intellector-NG

Os scripts de **'bootstrap'** para o Intellector-NG são simples, precisando apenas de pequenas modificações no que concerne a memória e variáveis de ambientes. Sugiro usar o nome *'run-intellector.sh'* no Linux permissão explícita de execução (básico para os linuxers!)

- no Linux/Unix; observe que coloco explicitamente a versão do JBoss; o ideal é criar um link simbólico e usar somente /opt/jboss e, também uso um IP 192.168.0.185, que provavelmente não será o seu, tar
- teremos mais abaixo, um arquivo de propriedades usadas pelo Intellector-NG e deve ficar em JBOSS_HOME/bin...

Não use a variável de ambiente **-Duser.dir=seu_dir**, pois o Intellector-NG para de compilar as políticas!

```
#!/bin/bash

# alguns snipets para o bootstrap do Intellector
export JAVA_OPTS="-Xms512m -Xmx536m -XX:PermSize=128m -XX:MaxPermSize=128m -Xss128k -Dintellector.datadir=/home/jboss/intellector"

# exportar a variavel JBOSS_HOME, devido a compilacao de politicas
export JBOSS_HOME=/opt/jboss-4.2.3.GA

# a library log4j deve estar incluido no classpath, inprescindivel na
# compilacao de politicas; simplemente não compila as politicas
export JBOSS_CLASSPATH=$JBOSS_HOME/server/default/lib/log4j.jar

# inicia o JBoss numa instancia e IP especificos
./run.sh -c default --host 192.168.0.185
```

- no Windows

```
rem alguns snipets para o bootstrap do Intellector
set JAVA_OPTS=-Xms512m -Xmx536m -XX:PermSize=128m -XX:MaxPermSize=128m -Xss128k -Dintellector.datadir=c:/java/intellector-files

rem exportar a variavel JBOSS_HOME, devido a compilacao de politicas
set JBOSS_HOME=c:\java\jboss-4.2.3.GA

rem a library log4j deve estar incluido no classpath, inprescindivel na
rem compilacao de politicas; simplemente não compila as politicas
set JBOSS_CLASSPATH=%JBOSS_HOME%\server\default\lib\log4j.jar

rem inicia o JBoss numa instancia e IP especificos
./run.bat -c default --host 192.168.0.185
```

Sem a linha **JBOSS_CLASSPATH=$JBOSS_HOME/server/default/lib/log4j.jar** no *bootstrap*, o Intellector-NG não compilará as políticas!

Área de dados para o Intellector-NG

O Intellector-NG tem uma característica onde, toda a persistência de dados é feita no *File System* da plataforma, seja persistência em XML (JAXB), seja dos plugins de acessos, políticas e layouts de políticas. verificar, que no arquivo de bootstrap, existe uma variável **intellector.datadir** que **deve** apontar para um diretório válido, onde ocorrerá a persistência e o site criará o restante dos diretórios; veja uma árvore de

Nunca MODIFIQUE NADA nos diretórios abaixo de **intellector.datadir**!

- considerando *intellector.datadir=/home/intellector* como diretório raiz...

1. /home/intellector/acessos - diretório de instalação dos plugins de acesso; o Intellector-NG cria uma pasta para cada acesso;
2. /home/intellector/key - diretório para as licenças do Intellector;
3. /home/intellector/dados - diretório raiz para persistência de cadastros (usuários, perfis, plugins, auditorias, etc.)
4. /home/intellector/dados/person - persistência de usuários cadastrados
5. /home/intellector/dados/audit - persistência de auditoria de execução de políticas (intellector.auditpolicy=true/false)
6. /home/intellector/dados/plugin - persistências dos dados dos plugins instalados
7. /home/intellector/dados/layouts - persistências de todos os layouts das políticas
8. /home/intellector/dados/policy - persistências de todos as políticas enviadas do servidor
9. /home/intellector/politicas - onde serão mantidas as classes das políticas compiladas
10. /home/intellector/politicas/classes
11. /home/intellector/politicas/classes/br
12. /home/intellector/politicas/classes/br/com
13. /home/intellector/politicas/classes/br/com/tools
14. /home/intellector/politicas/classes/br/com/tools/politicas

Instalando uma Licença para o Intellector

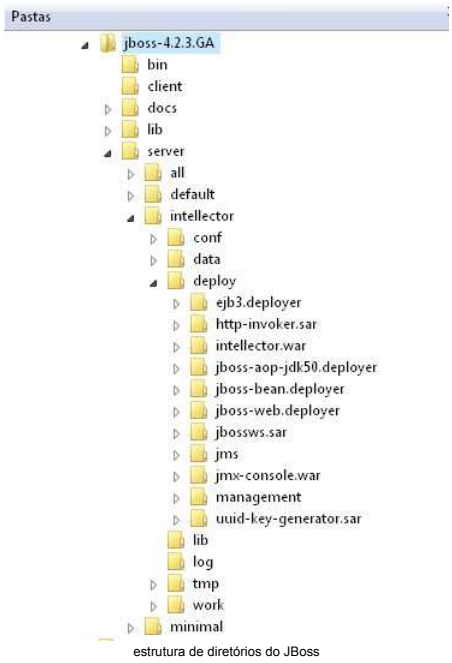
O Intellector trabalha com 3 (três) modelos de licenciamento, **enterprise, limited e trial**; essas licenças estarão contidas em um único arquivo nomeado **intellector.key** (outro nome não será reconhecido) que esse local a qualquer momento, não necessitando de qualquer intervenção no servidor.

- Tipos de licenças:

1. enterprise - totalmente livre para execução de qualquer política sem data de expiração
2. limited - determinada por um Tipo específico e uma data de expiração ou não
3. trial - Tipos de políticas livres e com uma data de expiração

Deployment do Intellector-NG no JBoss

Considerando que o item instalando o JBoss foi seguido, o próximo passo será o *deployment* (instalação) do pacote **intellector.war** em uma instância do JBoss. Nesse ponto, consideramos que a estrutura de



estrutura de diretórios do JBoss

Sigamos os passos...

- criar uma pasta **intellector.war** dentro do diretório *deploy*...
- extrair o pacote **intellector_V_xx_xxx_xxx_xxxxx.war** - somente seu conteúdo! - dentro do diretório recém-criado, intellector.war (os **X** são o controle de versões)
- se for uma atualização de versão, simplesmente remova o conteúdo da pasta **intellector.war, ANTES** de extrair a nova versão; é boa prática excluir as pastas log, tmp e work, situado abaixo da instância, e work; você não irá precisar deles.

Se for atualização, siga o sugerido acima.

saiba mais sobre estruturas de diretórios do JBoss.

Iniciando o JBoss

Se tudo foi bem, e os passos foram seguidos, iniciar o JBoss é a parte trivial; no diretório JBOSS_HOME/bin, inicie o *batch* (lembre-se que falamos dele aqui), simplesmente digitando **./run-intellector.sh** (linux

- iniciando...

```
=========================================================================

  JBoss Bootstrap Environment

  JBOSS_HOME: /opt/jboss-4.2.3.GA

  JAVA: /opt/jdk1.6.0_10/bin/java

  JAVA_OPTS: -Dprogram.name=run.sh -server -Xms512m -Xmx536m -XX:PermSize=128m -XX:MaxPermSize=128m -Xss128k -Dintellector.datadir=/home/jboss/intellecto

  CLASSPATH: /opt/jboss-4.2.3.GA/server/default/lib/log4j.jar:/opt/jboss-4.2.3.GA/bin/run.jar:/opt/jdk1.6.0_10/lib/tools.jar

=========================================================================

17:25:25,676 INFO  [Server] Starting JBoss (MX MicroKernel)...
17:25:25,708 INFO  [Server] Release ID: JBoss [Trinity] 4.2.3.GA (build: SVNTag=JBoss_4_2_3_GA date=200807181417)
17:25:25,710 INFO  [Server] Home Dir: /opt/jboss-4.2.3.GA
17:25:25,710 INFO  [Server] Home URL: file:/opt/jboss-4.2.3.GA/
```

- pronto pra atender requisições...

```
17:26:05,961 INFO  [intellector] ------------------------------------------------------------
17:26:05,962 INFO  [intellector] --------- Intellector configurado com sucesso... ----------
17:26:05,962 INFO  [intellector] ------------------------------------------------------------
17:26:06,146 INFO  [TomcatDeployer] deploy, ctxPath=/jmx-console, warUrl=.../deploy/jmx-console.war/
17:26:07,318 INFO  [Server] JBoss (MX MicroKernel) [4.2.3.GA (build: SVNTag=JBoss_4_2_3_GA date=200807181417)] Started in 21s:602ms
```

Configurações adicionais

Normalmente, ajustes adicionais, ou tuning, dependem de plataforma, número de requisições, escalabilidade, multi-ip, dentre outras; a mais comum é alterar a porta onde o *container* irá atender, *default* é a 808
analisadas e não fazem parte do escopo desse manual.
Configurando Pool de Conexões - JNDI

- Os datasources são configurados em arquivos xxx-ds.xml (onde xxx pode ser qualquer nome) que deverão ser inseridos no diretório **deploy** do servidor da aplicação. No exemplo usando JBOSS estão em

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- ===================================================================== -->
<!--                                                                       -->
<!--  JBoss Server Configuration                                           -->
<!--                                                                       -->
<!-- ===================================================================== -->

<!-- $Id: oracle-ds.xml 71535 2008-04-01 07:05:03Z adrian@jboss.org $ -->
```

```
<!-- ===================================================================== -->
<!--  Datasource config for Oracle originally from Steven Coy              -->
<!-- ===================================================================== -->

<datasources>
  <local-tx-datasource>
    <jndi-name>jdbc/OracleDS</jndi-name>
    <connection-url>jdbc:oracle:thin:@192.168.0.149:1521:oralin</connection-url>
        <!--
        Here are a couple of the possible OCI configurations.
        For more information, see [http://otn.oracle.com/docs/products/oracle9i/doc_library/release2/java.920/a96654/toc.htm]

        <connection-url>jdbc:oracle:oci:@youroracle-tns-name</connection-url>
                or
        <connection-url>jdbc:oracle:oci:@(description=(address=(host=youroraclehost)(protocol=tcp)(port=1521))(connect_data=(SERVICE_NAME=yourservicename)

        Clearly, its better to have TNS set up properly.
         -->
    <driver-class>oracle.jdbc.driver.OracleDriver</driver-class>
    <user-name>cartao_h</user-name>
    <password>uma_senha_qualquer</password>
    <!-- Uses the pingDatabase method to check a connection is still valid before handing it out from the pool -->
    <!--valid-connection-checker-class-name>org.jboss.resource.adapter.jdbc.vendor.OracleValidConnectionChecker</valid-connection-checker-class-name-->
    <!-- Checks the Oracle error codes and messages for fatal errors -->
    <exception-sorter-class-name>org.jboss.resource.adapter.jdbc.vendor.OracleExceptionSorter</exception-sorter-class-name>
    <!-- sql to call when connection is created
    <new-connection-sql>some arbitrary sql</new-connection-sql>
    -->

    <!-- sql to call on an existing pooled connection when it is obtained from pool - the OracleValidConnectionChecker is prefered
    <check-valid-connection-sql>some arbitrary sql</check-valid-connection-sql>
    -->

    <!-- corresponding type-mapping in the standardjbosscmp-jdbc.xml (optional) -->
    <metadata>
       <type-mapping>Oracle9i</type-mapping>
    </metadata>
  </local-tx-datasource>


</datasources>
```

- Baixe o Data Source para Oracle aqui (info).

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!-- ===================================================================== -->
<!--                                                                       -->
<!--  JBoss Server Configuration                                           -->
<!--                                                                       -->
<!-- ===================================================================== -->

<!-- $Id: sqlserver-ds.xml 71535 2008-04-01 07:05:03Z adrian@jboss.org $ -->
<!-- ===================================================================== -->
<!--  Datasource config for sqlserver originally from André Riba           -->
<!-- ===================================================================== -->
<datasources>
  <local-tx-datasource>
    <jndi-name>SqlServerDS</jndi-name>
    <connection-url>jdbc:jtds:sqlserver://192.168.0.85:1433/INDUSTRIAL_D</connection-url>
    <!-- <driver-class>com.microsoft.jdbc.sqlserver.SQLServerDriver</driver-class> -->
    <driver-class>net.sourceforge.jtds.jdbc.Driver</driver-class>
    <user-name>tools</user-name>
    <password>uma_senha_qualquer</password>
    <check-valid-connection-sql>select 1</check-valid-connection-sql>
    <min-pool-size>2</min-pool-size>
    <max-pool-size>10</max-pool-size>
        <!-- corresponding type-mapping in the standardjbosscmp-jdbc.xml (optional) -->
        <metadata>
          <type-mapping>MS SQLSERVER</type-mapping>
        </metadata>
  </local-tx-datasource>
</datasources>
```

- Baixe o Data Source para SqlServer aqui (info).

- Além disso, também deverão ser colocados no diretorio "lib" do JBOSS, os arquivos ".jar" referentes ao driver do banco de dados (ex : **C:\java\jboss-4.2.3.GA\server\default\lib**)

- Baixe o Driver JNDI para o SQLServer aqui.
- Baixe o Driver JNDI para o Oracle aqui.

Configuração jbossjca-service.xml

O *Intellector* utiliza a API de persistência de dados Hibernate de forma que no JBoss seja necessário alterar o arquivo jbossjca-service.xml na pasta **'deploy'** (ex : **C:\java\jboss-4.2.3.GA\server\default\deploy**

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- ===================================================================== -->
```

```
<!--                                                                      -->
<!--  JBoss JCA Configuration                                             -->
<!--                                                                      -->
<!-- ==================================================================== -->

<!-- $Id: jbossjca-service.xml 37786 2005-11-02 20:35:46Z adrian $ -->

<!--
   |  This contains configuration for the RARDeployer
      and some xsl based deployers.
 -->

<server>

  <!-- ==================================================================== -->
  <!-- JBossCX setup, for J2EE connector architecture support               -->
  <!-- The RARDeployer is needed only until xslt based deployment is written.-->
  <!-- ==================================================================== -->

  <mbean code="org.jboss.util.threadpool.BasicThreadPool"
         name="jboss.jca:service=WorkManagerThreadPool">
     <!-- The name that appears in thread names -->
     <attribute name="Name">WorkManager</attribute>
     <!-- The maximum amount of work in the queue -->
     <attribute name="MaximumQueueSize">1024</attribute>
     <!-- The maximum number of active threads -->
     <attribute name="MaximumPoolSize">100</attribute>
     <!-- How long to keep threads alive after their last work (default one minute) -->
     <attribute name="KeepAliveTime">60000</attribute>
  </mbean>

  <mbean code="org.jboss.resource.work.JBossWorkManager"
         name="jboss.jca:service=WorkManager">
     <depends optional-attribute-name="ThreadPoolName">jboss.jca:service=WorkManagerThreadPool</depends>
     <depends optional-attribute-name="XATerminatorName">jboss:service=TransactionManager</depends>
  </mbean>

  <mbean code="org.jboss.resource.deployment.RARDeployer"
              name="jboss.jca:service=RARDeployer">
           <depends optional-attribute-name="WorkManagerName">jboss.jca:service=WorkManager</depends>
     <depends optional-attribute-name="XATerminatorName">jboss:service=TransactionManager</depends>
  </mbean>

  <mbean code="org.jboss.deployment.XSLSubDeployer" name="jboss.jca:service=ConnectionFactoryDeployer">
    <attribute name="DdSuffix">-ds.xml</attribute>
    <attribute name="EnhancedSuffixes">300:-ds.xml</attribute>
    <attribute name="XslUrl">stylesheets/ConnectionFactoryTemplate.xsl</attribute>
    <attribute name="ValidateDTDs">false</attribute>
  </mbean>

  <!--
     | The CachedConnectionManager is used partly to relay started UserTransactions to
     | open connections so they may be enrolled in the new tx.
   -->
  <mbean code="org.jboss.resource.connectionmanager.CachedConnectionManager"
         name="jboss.jca:service=CachedConnectionManager">
    <depends optional-attribute-name="TransactionManagerServiceName">jboss:service=TransactionManager</depends>

    <!-- Enable connection close debug monitoring -->
    <!-- <attribute name="Debug">true</attribute> neste ponto pois a conexões com o banco de dados não serão gerenciadas pelo Servidor de Aplicação-->
    <attribute name="Debug">false</attribute>

  </mbean>

</server>
```

Ao configurar um arquivo de conexão com Banco de Dados no JBoss (JNDI), o Servidor de aplicação subentende que o **pool de conexões** será gerenciado por ele, podendo trazer instabilidade para as conexõ

```
2010-08-04 11:32:01,665 ERROR  [org.jboss.resource.connectionmanager.CachedConnectionManager] Closing a connection for you.  Please close them yourself: c
java.lang.Throwable: STACKTRACE
        at org.jboss.resource.connectionmanager.CachedConnectionManager.registerConnection(CachedConnectionManager.java:290)
        at org.jboss.resource.connectionmanager.BaseConnectionManager2.allocateConnection(BaseConnectionManager2.java:423)
        at org.jboss.resource.connectionmanager.BaseConnectionManager2$ConnectionManagerProxy.allocateConnection(BaseConnectionManager2.java:849)
        at org.jboss.resource.adapter.jdbc.WrapperDataSource.getConnection(WrapperDataSource.java:89)
        at org.hibernate.connection.DatasourceConnectionProvider.getConnection(DatasourceConnectionProvider.java:92)
        at org.hibernate.jdbc.ConnectionManager.openConnection(ConnectionManager.java:446)
        at org.hibernate.jdbc.ConnectionManager.getConnection(ConnectionManager.java:167)
        at org.hibernate.jdbc.JDBCContext.connection(JDBCContext.java:142)
        at org.hibernate.transaction.JDBCTransaction.begin(JDBCTransaction.java:85)
        at org.hibernate.impl.SessionImpl.beginTransaction(SessionImpl.java:1463)
        at br.com.totvs.persistence.factory.ConnectionFactory.beginTransaction(ConnectionFactory.java:194)
        at br.com.totvs.persistence.factory.ConnectionFactory.getSession(ConnectionFactory.java:165)
        at br.com.tools.intellector.api.persistence.PersistenceUtil.expungeDatabase(PersistenceUtil.java:87)
        at br.com.tools.intellector.action.ExecINTExpurgarDadosAcessosAction.doEfetuaExpurgo(ExecINTExpurgarDadosAcessosAction.java:113)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
        at java.lang.reflect.Method.invoke(Method.java:597)
        at com.toolssoftware.framework.portal.PortalAction.process(PortalAction.java:82)
        at com.toolssoftware.framework.struts.action.Action.execute(Action.java:229)
        at org.apache.struts.action.RequestProcessor.processActionPerform(RequestProcessor.java:425)
```

```
        at org.apache.struts.action.RequestProcessor.process(RequestProcessor.java:228)
        at org.apache.struts.action.ActionServlet.process(ActionServlet.java:1913)
        at com.toolssoftware.framework.struts.action.ActionServlet.process(ActionServlet.java:98)
        at org.apache.struts.action.ActionServlet.doPost(ActionServlet.java:462)
        at javax.servlet.http.HttpServlet.service(HttpServlet.java:710)
        at javax.servlet.http.HttpServlet.service(HttpServlet.java:803)
        at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:290)
        at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
        at br.com.tools.intellector.filter.SecurityFilter.doFilter(SecurityFilter.java:56)
        at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:235)
        at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
        at org.jboss.web.tomcat.filters.ReplyHeaderFilter.doFilter(ReplyHeaderFilter.java:96)
        at org.apache.catalina.core.ApplicationFilterChain.internalDoFilter(ApplicationFilterChain.java:235)
        at org.apache.catalina.core.ApplicationFilterChain.doFilter(ApplicationFilterChain.java:206)
        at org.apache.catalina.core.StandardWrapperValve.invoke(StandardWrapperValve.java:230)
        at org.apache.catalina.core.StandardContextValve.invoke(StandardContextValve.java:175)
        at org.jboss.web.tomcat.security.SecurityAssociationValve.invoke(SecurityAssociationValve.java:182)
        at org.jboss.web.tomcat.security.JaccContextValve.invoke(JaccContextValve.java:84)
        at org.apache.catalina.core.StandardHostValve.invoke(StandardHostValve.java:127)
        at org.apache.catalina.valves.ErrorReportValve.invoke(ErrorReportValve.java:102)
        at org.jboss.web.tomcat.service.jca.CachedConnectionValve.invoke(CachedConnectionValve.java:157)
        at org.apache.catalina.core.StandardEngineValve.invoke(StandardEngineValve.java:109)
        at org.apache.catalina.connector.CoyoteAdapter.service(CoyoteAdapter.java:262)
        at org.apache.coyote.http11.Http11Processor.process(Http11Processor.java:844)
        at org.apache.coyote.http11.Http11Protocol$Http11ConnectionHandler.process(Http11Protocol.java:583)
        at org.apache.tomcat.util.net.JIoEndpoint$Worker.run(JIoEndpoint.java:446)
        at java.lang.Thread.run(Thread.java:619)
```

Intellector-EAR

O *Intellector-EAR* é a produto de persistência do Intellector Server, monitorando um fila (queue) no Servidor de Aplicação onde os dados das execuções de políticas e plugins de acessos é enviada para serem

Para instalar o Intellector-EAR:
No JBoss: Basta colocar na pasta deploy (ex : **C:\java\jboss-4.2.3.GA\server\default\deploy**) do Servidor de Aplicação.

Baixe o Intellector-EAR aqui (info) .

Configurando a "fila" do Intellector

JBoss

No Servidor de aplicação na pasta "(ex : **C:\java\jboss-4.2.3.GA\server\default\deploy\jms\jbossmq-destinations-service.xml)", neste arquivo contém as configurações das filas que o servidor disponi**

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- $Id: jbossmq-destinations-service.xml 25907 2004-11-16 04:32:39Z ejort $ -->

<!--
   | This file defines the default Queues and Topics that JBossMQ
   | ships with.  The default Queues and Topics are used by the
   | JBoss test suite and by the sample jms programs.
   |
   | You can add other destinations to this file, or you can create other
   | *-service.xml files to contain your application's destinations.
 -->

<server>
  <!-- Destination without a configured SecurityManager or without a
       a SecurityConf will default to role guest with read=true, write=true,
       create=false.
  -->
  <mbean code="org.jboss.mq.server.jmx.Topic"
         name="jboss.mq.destination:service=Topic,name=testTopic">
    <depends optional-attribute-name="DestinationManager">jboss.mq:service=DestinationManager</depends>
    <depends optional-attribute-name="SecurityManager">jboss.mq:service=SecurityManager</depends>
    <attribute name="SecurityConf">
      <security>
        <role name="guest" read="true" write="true"/>
        <role name="publisher" read="true" write="true" create="false"/>
        <role name="durpublisher" read="true" write="true" create="true"/>
      </security>
    </attribute>
  </mbean>

  <mbean code="org.jboss.mq.server.jmx.Topic"
         name="jboss.mq.destination:service=Topic,name=securedTopic">
    <depends optional-attribute-name="DestinationManager">jboss.mq:service=DestinationManager</depends>
    <depends optional-attribute-name="SecurityManager">jboss.mq:service=SecurityManager</depends>
    <attribute name="SecurityConf">
      <security>
        <role name="publisher" read="true" write="true" create="false"/>
      </security>
```

```
    </attribute>
  </mbean>

  <mbean code="org.jboss.mq.server.jmx.Topic"
         name="jboss.mq.destination:service=Topic,name=testDurableTopic">
    <depends optional-attribute-name="DestinationManager">jboss.mq:service=DestinationManager</depends>
    <depends optional-attribute-name="SecurityManager">jboss.mq:service=SecurityManager</depends>
    <attribute name="SecurityConf">
       <security>
         <role name="guest" read="true" write="true"/>
         <role name="publisher" read="true" write="true" create="false"/>
         <role name="durpublisher" read="true" write="true" create="true"/>
       </security>
    </attribute>
  </mbean>

  <mbean code="org.jboss.mq.server.jmx.Queue"
         name="jboss.mq.destination:service=Queue,name=testQueue">
    <depends optional-attribute-name="DestinationManager">jboss.mq:service=DestinationManager</depends>
    <depends optional-attribute-name="SecurityManager">jboss.mq:service=SecurityManager</depends>
    <attribute name="MessageCounterHistoryDayLimit">-1</attribute>
    <attribute name="SecurityConf">
      <security>
        <role name="guest" read="true" write="true"/>
        <role name="publisher" read="true" write="true" create="false"/>
        <role name="noacc" read="false" write="false" create="false"/>
      </security>
    </attribute>
  </mbean>
  <mbean code="org.jboss.mq.server.jmx.Queue"
         name="jboss.mq.destination:service=Queue,name=A">
    <depends optional-attribute-name="DestinationManager">jboss.mq:service=DestinationManager</depends>
  </mbean>
  <mbean code="org.jboss.mq.server.jmx.Queue"
         name="jboss.mq.destination:service=Queue,name=B">
    <depends optional-attribute-name="DestinationManager">jboss.mq:service=DestinationManager</depends>
  </mbean>
  <mbean code="org.jboss.mq.server.jmx.Queue"
         name="jboss.mq.destination:service=Queue,name=C">
         <depends optional-attribute-name="DestinationManager">jboss.mq:service=DestinationManager</depends>
  </mbean>
  <mbean code="org.jboss.mq.server.jmx.Queue"
         name="jboss.mq.destination:service=Queue,name=D">
    <depends optional-attribute-name="DestinationManager">jboss.mq:service=DestinationManager</depends>
  </mbean>
  <mbean code="org.jboss.mq.server.jmx.Queue"
         name="jboss.mq.destination:service=Queue,name=ex">
    <depends optional-attribute-name="DestinationManager">jboss.mq:service=DestinationManager</depends>
  </mbean>
  <mbean code="org.jboss.mq.server.jmx.Queue"
         name="jboss.mq.destination:service=Queue,name=intellector">
    <depends optional-attribute-name="DestinationManager">jboss.mq:service=DestinationManager</depends>
  </mbean>

</server>
```

Neste arquivo somente deve ser incluído a fila do Intelletor caso o Administrador do Servidor de Aplicação deseje que o Intellector não ocupe uma fila que poderá ser utilizada por outras aplicações. Neste caso

```
 <mbean code="org.jboss.mq.server.jmx.Queue"
         name="jboss.mq.destination:service=Queue,name=intellector">
    <depends optional-attribute-name="DestinationManager">jboss.mq:service=DestinationManager</depends>
  </mbean>
```

para que o Intellector possa monitorar uma fila própria. Para configurar a Fila que o Intellector irá monitorar o Intellector possui um arquivo de configuração para as persistências de dados explicado mais adiant

pesistence.properties

Para utilização da persistencia de acessos no *'intellector Server*', na primeira execução antes da utilização do cache, deverá ser copiado o arquivo de configuração **persistence.properties** para o diretório $JB(
O Intellector ultiliza a API de persistência *Hibernate* de forma que este arquivo também serve para mapear as configurações mais eficazes para utilização da API.

```
#persistence.properties
#Fri Feb 03 13:52:26 BRST 2012

persistence.queue.name=queue/intellector
persistence.queue.connection.factory=QueueConnectionFactory

persistence.jndi.context.factory=org.jboss.security.jndi.JndiLoginInitialContextFactory
persistence.jndi.address=jnp://127.0.0.1\:1099
persistence.jndi.security.principal=
persistence.jndi.security.credentials=

hibernate.connection.driver_class=net.sourceforge.jtds.jdbc.Driver
hibernate.connection.url=jdbc:jtds:sqlserver://192.168.0.242:1433/totvs_d_esquema
#hibernate.default_schema=
hibernate.connection.username=tools
hibernate.connection.password=toolsscc
hibernate.transaction.factory_class=org.hibernate.transaction.JDBCTransactionFactory
hibernate.dialect=org.hibernate.dialect.SQLServerDialect

#hibernate.connection.datasource=java\:jdbc/tools
```

```
hibernate.jdbc.batch_versioned_data=true
hibernate.cache.use_second_level_cache=false
hibernate.show_sql=true
hibernate.transaction.flush_before_completion=true
hibernate.generate_statistics=true
hibernate.hbm2ddl.auto=update
hibernate.cache.use_query_cache=false
hibernate.max_fetch_depth=1
hibernate.connection.release_mode=after_statement
hibernate.format_sql=true
hibernate.session_factory_name=sessionFactory
hibernate.transaction.auto_close_session=true
hibernate.use_sql_comments=true
hibernate.hbm2ddl.delimiter=;
intellector.usesdatabase=true
hibernate.connection.autocommit=false
hibernate.cache.provider_class=org.hibernate.cache.NoCacheProvider
hibernate.current_session_context_class=thread
```

- persistence.jndi.address=Endereço de conexão para a fila do Intellector.
- persistence.jndi.security.principal= Usuário de conexão com o Servidor de Aplicação (Configuração específica do Servidor de Aplicação)
- persistence.jndi.security.credentials= Senha de conexão com o Servidor de Aplicação (Configuração específica do Servidor de Aplicação)
- persistence.queue.name= Fila para qual será enviada as mensagens (dados de execução) do Intellector.
- persistence.queue.connection.factory= Fábrica de Conexão das Filas para envio de mensagens.
- persistence.jndi.context.factory= Fábrica de criação de contexto.(Configuração específica para criação da Fila do Intellector)
- hibernate.jdbc.batch_versioned_data= Batch para criação de datas no banco de dados.
- hibernate.cache.use_second_level_cache= Usa o segundo level de cache para consultas no banco de dados.
- hibernate.show_sql= Mostra os comandos 'sql' no log da aplicação.
- hibernate.connection.datasource= Conexão 'JNDI' utilizada para persistência de dados.
- hibernate.generate_statistics= Gera estátisticas do Banco de Dados (Configuração específica para o Hibernate).
- hibernate.hbm2ddl.auto= Esquema de criação e atualização de dados do Banco de Dados (Configuração específica para o Hibernate).
- hibernate.cache.use_query_cache= Utiliza cache para as consultas no banco de dados (Configuração específica para o Hibernate).
- hibernate.max_fetch_depth= Máximo de 'fetch' (Configuração específica para o Hibernate).
- hibernate.format_sql= Formata os comando 'sql' (Configuração específica para o Hibernate).
- hibernate.use_sql_comments= No Log da aplicação são colocados comentários para cada comando 'sql' executado (Configuração específica para o Hibernate).
- hibernate.hbm2ddl.delimiter= Delimitador para o Banco de Dados (Configuração específica para o Hibernate).
- hibernate.dialect= Dialeto utlizado para conexão com o Banco de Dados.

Baixe aqui (info) um exemplo do persistence.properties

Deverão ser colocados no diretorio "lib" do JBOSS, os arquivos ".jar" referentes ao driver do banco de dados (ex : **C:\java\jboss-4.2.3.GA\server\default\lib**)

503 Views, 5 Attachments

Add Child Page

Comments

Post Reply